Application de la théorie des nombres à la conception optimale et à l'implémentation de très faible complexité des filtres numériques

Ali Daher

THÈSE / UNIVERSITÉ DE BRETAGNE OCCIDENTALE

sous le sceau de l'Université européenne de Bretagne

pour obtenir le titre de

DOCTEUR DE L'UNIVERSITÉ DE BRETAGNE OCCIDENTALE

Mention : Sciences et Technologies de l'Information et de la Communication - Spécialité Traitement du Signal et des Images

École Doctorale SICMA - ED 373

présentée par

Ali DAHER

Préparée au Lab-STICC (UMR CNRS 3192)

Application de la théorie des nombres à la conception optimale et à l'implémentation de très faible complexité des filtres numériques

Thèse soutenue le 8 décembre 2009

devant le jury composé de :

Olivier SENTIEYS
Professeur, IRISA, ENSSAT Lannion / *Président du jury*

Jean-François HELARD
Professeur, IETR, INSA Rennes / *Rapporteur*

Daniel ROVIRAS
Professeur, Chaire des Systèmes de Télécommunications, CNAM Paris / *Rapporteur*

Pascal CHARGE
Maître de Conférences, HDR, LATTIS, INSA Toulouse / *Examinateur*

El-Houssaïn BAGHIOUS
Maître de Conférences, Lab-STICC, UBO / *Examinateur*

Gilles BUREL
Professeur, Lab-STICC, UBO / *Codirecteur de thèse*

Emanuel RADOI
Professeur, Lab-STICC, UBO / *Codirecteur de thèse*

Roland GAUTIER
Maître de Conférences, Lab-STICC, UBO / *Invité*

École Doctorale SICMA [ED 373]
(Santé, Information-Communications, Mathématiques, Matière)

THÈSE DE DOCTORAT
DE
L'UNIVERSITÉ DE BRETAGNE OCCIDENTALE

Mention "Sciences et Technologies de l'Information et de la Communication"

Spécialité "Traitement du Signal et des Images"

Présentée par

Ali DAHER

Préparée au

Lab-STICC (UMR CNRS 3192)

Application de la théorie des nombres à la conception optimale et à l'implémentation de très faible complexité des filtres numériques

Soutenue le 8 Décembre 2009 devant le jury composé des membres ci-dessous désignés :

Président du jury

M.	Olivier SENTIEYS	Professeur, IRISA, ENSSAT Lannion

Rapporteurs

M.	Jean-François HELARD	Professeur, IETR, INSA Rennes
M.	Daniel ROVIRAS	Professeur, Chaire des Systèmes de Télécommunications, CNAM Paris

Examinateurs

M.	El Houssaïn BAGHIOUS	Maître de Conférences, Lab-STICC, UBO
M.	Gilles BUREL	Professeur, Lab-STICC, UBO / Codirecteur de thèse
M.	Pascal CHARGE	Maître de Conférences, HDR, LATTIS, INSA Toulouse
M.	Emanuel RADOI	Professeur, Lab-STICC, UBO / Codirecteur de thèse

Invité

M.	Roland GAUTIER	Maître de Conférences, Lab-STICC, UBO

Remerciements

Le travail présenté dans ce mémoire a été réalisé au sein du Laboratoire des Sciences et Technologies de l'Information, de la Communication et de la Connaissance (Lab-STICC) qui est une unité mixte de recherche entre le CNRS (UMR CNRS 3192) et l'Université Européenne de Bretagne dont l'Université de Bretagne Occidentale (UBO) fait partie.

Je tiens à exprimer ma profonde gratitude à Monsieur Gilles BUREL et Monsieur Emanuel RADOI, Professeurs à l'Université de Bretagne Occidentale (UBO) et directeurs de thèse, pour l'aide et le soutien qu'il m'ont apporté tout au long de ce travail. Leur grande culture scientifique et leurs qualités humaines sont autant d'éléments qui m'ont conforté tout au long de l'avancement de mes travaux.

Je tiens également à exprimer ma reconnaissance à Monsieur El-Houssaïn BAGHIOUS, Maître de Conférences à l'UBO, pour ses conseils pédagogiques, sa bonne humeur et pour avoir encadré ce travail de thèse.

Je tiens à remercier Monsieur Olivier SENTIEYS, Professeur à l'Ecole Nationale Supérieure des Sciences Appliquées et de Technologie (ENSSAT) de Lannion pour avoir accepté la présidence du jury de cette thèse.

Je remercie Monsieur Jean-François HELARD, Professeur à l'Institut National des Sciences Appliquées (INSA) de Rennes, et Monsieur Daniel ROVIRAS, Professeur, titulaire de la Chaire des Systèmes de Télécommunications au CNAM Paris, pour avoir accepté d'être les rapporteurs de ce travail.

Je remercie également Monsieur Pascal CHARGE, Maître de Conférences à l'Institut National des Sciences Appliquées (INSA) de Toulouse, et Monsieur Roland GAUTIER, Maître de Conférences à l'UBO, pour l'intérêt qu'ils ont porté à cette étude en acceptant de faire partie de ce jury.

Enfin, je dédie ce travail à mes parents, mes frères et ma soeur, et à tous ceux qui m'ont soutenu et supporté durant ces trois années.

Table des matières

Table des figures

Liste des tableaux

Liste des acronymes et abrévitions

RIF	Réponse Impulsionnelle Finie
RII	Réponse Impulsionnelle Infinie
OLS	OverLap-Save
OLA	OverLap-Add
BDF	Block Digital Filtering
TFD	Transformée de Fourier Discrète
TFDI	Transformée de Fourier Discrète Inverse
TFR	Transformée de Fourier Rapide
DSP	Digital Signal Processing
EQM	Erreur Quadratique Moyenne
FNT	Fermat Number Transform
IFNT	Inverse **FNT**
GF	Galois Field
ILS	Integer Least Squares
LS	Least Squares
LMS	Least Mean Square

Notations Mathématiques

A^T	Transposé du vecteur ou de la matrice A
$A^\star$	Transposée conjuguée de A
$A(:,j)$	Vecteur formé par les éléments de la $j^{ème}$ colonne de la matrice A
$A(i,:)^T$	Vecteur formé par les éléments de la $i^{ème}$ ligne de la matrice A
$A(i,j)$	Elément de la $i^{ème}$ ligne et de la $j^{ème}$ colonne de la matrice$A$
$pgcd$	Plus Grand Commun Diviseur
$round(a)$	Entier le plus proche de a
$\langle . \rangle_q$	Opération modulo q
$\lvert . \rvert$	Module d'un nombre complexe
$a \bmod q$ ou $\langle a \rangle_q$	Réduction de a par le modulo q
$\mathbb{Z}_q$	Ensemble des entiers appartenant à $\{0, q-1\}$
$GF(q)$	Corps de Galois d'ordre q
$\equiv$	Congruence
F_t	Nombre de Fermat d'ordre t
$*$	Produit de convolution linéaire
$\circ$	Produit de convolution circulaire
$\bullet$	Multiplications terme à terme de deux vecteurs
$\boxdot$	Multiplication terme à terme de deux matrices (produit d'Hadamard)

Introduction

Malgré le progrès de la microélectronique des circuits VLSI (Very Large Scale Integration) et le développement des processeurs DSP (Digital Signal Processing), la complexité et les erreurs restent des facteurs cruciaux pour le développement des systèmes de télécommunications, surtout lorsqu'il s'agit des applications de grande précision en temps réel. Ces applications nécessitent plusieurs opérations de pré-traitement dont celle qui consiste à transformer le signal récupéré pour en extraire l'information utile. Cette opération, souvent préalable à tout traitement, est celle du filtrage fréquentiel.

Dans le domaine numérique, les filtres sont classés en deux catégories : les filtres à réponse impulsionnelle finie (RIF, en anglais FIR : Finite Impulse Response) et ceux à réponse impulsionnelle infinie (RII, en anglais IIR : Infinite Impulse Response). Un intérêt particulier est porté essentiellement sur le filtrage RIF en raison des différents avantages qu'il offre. Dans des domaines tels que les télécommunications, ce filtrage RIF nécessite des filtres de grande longueur impliquant une charge importante pour le calcul de convolution. Le coût de filtrage est essentiellement dû aux multiplications qui sont plus coûteuses en temps de calcul que les additions lors de leur implantation sur des processeurs DSP. Pour réduire ce coût, plusieurs algorithmes, qui réduisent théoriquement le nombre de multiplications mais qui s'avèrent très difficiles à être implantés, ont été proposés [Winograd1980]. L'algorithme le plus efficace pour un calcul rapide des convolutions linéaires reste celui de la convolution cyclique qui utilise comme moyen de calcul intermédiaire la Transformée de Fourier Rapide (TFR, en anglais FFT : Fast Fourier Transform), introduite par Cooley et Tukey [Cooley1965]. Deux méthodes de filtrage RIF par blocs sont bien connues et utilisées, l'overlap-save (OLS) et l'overlap-add (OLA). Ces deux méthodes de filtrage indirect consistent à traiter le signal par blocs au moyen de la TFR permettant ainsi de réduire la complexité arithmétique des

calculs de convolution. Comparé à l'algorithme de calcul direct de la convolution, elles présentent un gain appréciable notamment pour les filtres RIF de grande longueur. En outre, les différents blocs peuvent être traités en parallèle par plusieurs processeurs ou unités de calcul, ce qui permet de réduire davantage le temps global de traitement.

Alors que les filtres RIF associés aux méthodes OLS et OLA sont souvent synthétisés par les méthodes classiques de synthèse des filtres RIF, nous développerons dans ce mémoire des nouvelles approches de conception optimale de ces filtres adaptées à la structure bien particulière de l'OLS et de l'OLA. Le critère d'optimisation choisi est celui de la minimisation de l'erreur quadratique moyenne (distorsion) entre le filtrage désiré et le filtrage par blocs obtenu. Nous proposerons ensuite une nouvelle conception dite généralisée du filtrage par blocs permettant de réaliser un compromis entre la distorsion et le coût du filtrage par blocs.

Bien que les calculs présents dans les algorithmes de filtrage ne présentent aucune contrainte pour leur implantation sur des processeurs à virgule flottante, des erreurs dues à la quantification des données et à l'arrondi des calculs apparaissent lorsqu'il s'agit d'une implémentation en virgule fixe. Toutefois, dans de nombreuses applications en télécommunications, une conception basée sur une unité de calcul en virgule fixe est favorisée. En effet, la complexité des calculs et, par conséquent, la taille de la puce et le coût d'un microcontrôleur sont fortement accrus quand il s'agit d'une conception avec une unité de calcul en virgule flottante. Ainsi, nous proposons une nouvelle approche optimale de conception du filtrage OLS et OLA mettant en oeuvre la transformée en nombres de Fermat (FNT : Fermat Number Transform). Cette transformée, définie sur un corps de Galois d'ordre égal à un nombre de Fermat, est un cas particulier de la transformée en nombres entiers (NTT : Number Theoretic Transform). En virgule fixe, elle permet un calcul sans erreur ainsi qu'une large réduction du nombre de multiplications nécessaires à la réalisation du produit de convolution.

Ce rapport de thèse, réalisée au sein du Laboratoire en Sciences et Technologies de l'Information, de la Communication et de la Connaissance (Lab-STICC) UMR CNRS 3192, présente notre travail portant sur la conception optimale des filtres numériques RIF de très faible complexité. Notre étude a été menée pour améliorer les performances et pour diminuer la complexité du filtrage RIF basé sur les méthodes de filtrage par blocs overlap-save et overlap-add. L'étude principale consiste à proposer et à développer de nouveaux algorithmes de faible complexité pour une conception optimale de ces filtres RIF. Les trois principaux facteurs étudiés sont : la complexité du filtrage, les erreurs de distorsion du filtrage et la complexité des algorithmes de synthèse des filtres RIF.

Ce rapport de thèse comprend cinq chapitres :

Le premier rappelle les notions fondamentales du filtrage numérique et les méthodes classiques de synthèse des filtres RIF. Le principe et l'intérêt des méthodes de filtrage par blocs overlap-save (OLS) et overlap-add (OLA) sont aussi décrits.

Dans le deuxième chapitre, nous abordons le problème d'optimisation des filtres RIF pour une implantation selon les méthodes OLS et OLA. Nous développons les structures matricielles du filtrage désiré et du filtrage par blocs OLS et OLA. Nous proposons ensuite une nouvelle approche de synthèse optimale de ces filtres blocs. Le critère d'optimisation choisi est celui de la minimisation de l'erreur quadratique moyenne (distorsion) entre le filtrage désiré et le filtrage par blocs. En nous basant sur les propriétés des matrices circulantes, nous développons un algorithme de très faible complexité de synthèse de ces filtres blocs optimaux. Une étude comparative entre les deux méthodes optimales OLS et OLA est ensuite réalisée. Cette comparaison démontre que ces méthodes diffèrent sur la distribution fréquentielle de l'erreur de repliement de spectre (aliasing) lorsque celle-ci existe.

Dans le troisième chapitre, nous décrivons une nouvelle conception, dite généralisée, du filtrage par blocs dans laquelle le filtre bloc est représenté par une matrice et non plus par un vecteur comme c'est dans le cas d'un filtre classique. Cette nouvelle conception permettra de réaliser un compromis entre la distorsion et le coût du filtrage par blocs. Nous développons quatre méthodes de synthèse du filtre bloc optimal dont l'utilisation de l'une ou de l'autre dépend du type de la transformée utilisée (unitaire, non unitaire) et de la pondération affectée aux fréquences (pondéré ou non pondéré).

Le quatrième chapitre est une introduction aux transformées en nombres entiers (NTT). Nous y présentons la définition et les propriétés fondamentales de ces transformées. Le domaine de définition d'une NTT étant inclus dans celui des nombres entiers, son utilisation peut favoriser des implantations sur des processeurs DSP à virgule fixe. La transformée en nombres de Fermat (FNT : Fermat Number Transform), définie sur un corps de Galois d'ordre égal à un nombre de Fermat, est un cas particulier de la transformée en nombres entiers (NTT : Number Theoretic Transform). En virgule fixe, cette transformée permet un calcul sans erreur ainsi qu'une large réduction du nombre de multiplications nécessaires à la réalisation du produit de convolution circulaire.

Dans le dernier chapitre, nous nous sommes confrontés aux problèmes d'implantation efficace des algorithmes OLS et OLA sur des DSPs à virgule fixe. Le choix d'un processeur à virgule fixe provient du fait que son coût de calcul et sa consommation sont largement réduits par rapport à un processeur à virgule flottante. Cependant, le calcul en virgule fixe induit sur les données des effets inévitables

de troncature et d'arrondi qui se traduisent par du bruit de calcul. Pour réduire ces erreurs et aussi la complexité de ces algorithmes, nous proposons dans ce chapitre une nouvelle implantation des techniques OLS et OLA mettant en oeuvre la FNT qui remplace la TFD. Nous sommes ensuite confrontés au problème de synthèse du filtre bloc optimal pour cette nouvelle implantation. Un problème des moindres carrés en nombres entiers avec une restriction des solutions sur le corps de Galois sera posé puis étudié. A part la méthode triviale de recherche exhaustive de grande complexité, les méthodes d'optimisation s'avèrent inadaptées à la résolution de notre problème. Pour apporter une solution à ce problème, nous développons une nouvelle propriété de diagonalisation des matrices circulantes faisant appel à la FNT et à des calculs définis sur le corps de Galois. A partir de cette nouvelle propriété, un algorithme très efficace pour la synthèse du filtre bloc FNT optimal est développé. Ce nouvel algorithme associé à la mise en oeuvre de la FNT dans le filtrage OLS et OLA permet une implantation du filtrage par blocs sur des processeurs à virgule fixe avec des erreurs de distorsion et une complexité de filtrage plus réduites que celles obtenues dans le cas d'une implantation au moyen de la TFD.

Enfin, une dernière partie conclut ce travail de thèse et expose quelques perspectives envisageables qui peuvent compléter le travail.

Chapitre 1

Filtrage à réponse impulsionnelle finie

1.1 Introduction

Avec le développement des systèmes de télécommunications numériques, de nombreuses applications nécessitant une grande charge de calcul apparaissent. Ainsi, des problèmes de temps et de complexité de calcul se posent lorsque nous cherchons à réaliser une implantation temps réel des algorithmes associés à ces applications. Le filtrage à réponse impulsionnelle finie (RIF) qui occupe une place fondamentale dans ce domaine de traitement numérique des signaux représente une grande partie de cette charge de calcul.

Pour réduire la complexité du filtrage RIF, des algorithmes de filtrage rapide ont été proposés dans la littérature. Parmi eux, les algorithmes de filtrage par blocs restent les plus utilisés. Dans ce chapitre, nous nous intéressons aux deux algorithmes de filtrage RIF par blocs : l'overlap-save et l'overlap-add. Nous rappelons quelques notions de base sur le principe et les méthodes classiques de synthèse des filtres RIF. Nous expliquons ensuite le principe des techniques overlap-save et overlap-add basées sur l'algorithme de la convolution cyclique et utilisant comme moyen de calcul intermédiaire la transformée de Fourier rapide (TFR). Nous montrerons leur intérêt en comparant leur complexité à celle de la réalisation directe de filtrage. Cet intérêt réside dans une réduction de la complexité arithmétique du filtrage permettant ainsi d'obtenir des gains appréciables surtout pour les filtres de grande longueur.

Bien que les processeurs récents aient des puissances de calcul de plus en plus importantes, les multiplications restent bien plus complexes qu'une réalisation d'additions ou de décalage de bits. C'est pourquoi nous nous intéressons en particulier au nombre de multiplications pour l'évaluation

du coût des différents algorithmes. Cette évaluation ne sera donnée qu'à titre indicatif et ne représentera en aucun cas le coût de calcul réel pour l'implantation des différents algorithmes. En se référant à la méthode de Golub (Golub's method) [Golub1989] pour la multiplication des complexes, nous considérons qu'une multiplication dans le domaine des complexes est équivalente à trois multiplications réelles.

1.2 Classification des filtres

Le principe de filtrage fréquentiel est de modifier le spectre d'un signal en favorisant certaines fréquences et en atténuant d'autres. Ainsi, un filtre numérique de réponse impulsionnelle $h(n)$ et de spectre $\overline{h}(f)$, appliqué à un signal numérique d'entrée $x(n)$ de spectre $\overline{x}(f)$, donne en sortie un signal numérique $y(n)$ dont le spectre $\overline{y}(f)$ est donné par :

$$\overline{y}(f) = \overline{x}(f).\overline{h}(f) \tag{1.1}$$

Cette équation se traduit dans le domaine temporel par :

$$y(n) = x(n) * h(n) \tag{1.2}$$

où $*$ correspond à un produit de convolution linéaire. Ainsi la sortie $y(n)$ peut s'écrire :

$$y(n) = \sum_{m} h(m)\, x(n-m) \tag{1.3}$$

Nous distinguons deux grandes familles de filtres numériques : les filtres à réponse impulsionnelle infinie RII (IIR, en anglais : Infinite Impulse Response) et les filtres à réponse impulsionnelle finie RIF (FIR en anglais : Finite Impulse Response) [Parks1987, Oppenheim1989].

Les filtres RII et leurs méthodes de synthèse sont inspirés des techniques de filtrage analogique traditionnelles : Butterworth, Tchebychev, Bessel, Elliptique, etc. Le problème qui consiste à transformer un filtre analogique en un filtre numérique est relativement simple. De ce fait, de nombreuses méthodes sont proposées pour concevoir un filtre numérique RII.

Quant aux filtres RIF, ils présentent l'inconvénient de nécessiter une grande charge de calcul, comparée à celle exigée par les filtres RII [Rabiner1971]. Malgré cette contrainte, les performances des processeurs actuellement disponibles rendent ces filtres RIF largement utilisés, puisqu'ils présentent

des méthodes de synthèse permettant d'obtenir des filtres toujours stables et à phase linéaire, quelle que soit la réponse fréquentielle désirée.

Les filtres RIF sont définis par leur réponse impulsionnelle $h(n)$ de taille finie, soit :

$$h(n) = 0 \ \ si \ \ n < 0 \ \ ou \ \ n > P \tag{1.4}$$

où P désigne l'ordre du filtre. Ainsi, le produit de convolution de la relation (1.3) s'écrit :

$$y(n) = \sum_{m=0}^{P} h(m)\, x(n-m) \tag{1.5}$$

Afin de pouvoir comparer par la suite les méthodes de synthèse proposées dans notre étude aux méthodes classiques de synthèse des filtres RIF, nous présentons tout d'abord le principe général de ces dernières.

1.3 Méthodes classiques de synthèse des filtres RIF

Les filtres RIF, appelés aussi filtres non récursifs, permettent de réaliser le filtrage numérique au moyen du produit de convolution défini par la relation (1.5) : les valeurs de sortie $y(n)$ sont obtenues par une somme pondérée d'un ensemble fini de valeurs d'entrée $x(n)$ représentant les échantillons du signal à filtrer. Quant aux coefficients de pondération $h(n)$, ils sont obtenus selon l'une des méthodes de synthèse des filtres RIF développées dans la littérature [Rabiner1975, Oppenheim1975]. Ces méthodes, développées sous forme d'algorithmes de synthèse, comportent généralement quatre étapes :

- choix du filtre désiré, souvent donné par son spectre ou sa réponse en fréquence,
- choix du type du filtre à concevoir, RIF ou RII. Dans notre étude, nous nous limitons au cas des filtres RIF,
- choix d'un critère d'évaluation du filtre à obtenir par rapport au filtre désiré,
- développement d'une méthode pour obtenir le filtre optimal répondant au critère.

Les méthodes de synthèse des filtres RIF peuvent être regroupées en trois classes :

1. la méthode des fenêtres ;
2. la méthode de l'échantillonnage en fréquence ;
3. les méthodes d'optimisation.

1.3.1 Méthode des fenêtres

La réponse impulsionnelle $h_d(n)$ associée à un filtre désiré donné par sa réponse en fréquence $\overline{h_d}(f)$ peut être obtenue soit par un calcul de transformée de Fourier inverse à temps discret selon l'équation (1.6), soit par un calcul de transformée de Fourier inverse selon l'équation (1.7) suivi d'un échantillonnage de la fonction $h_d(t)$ obtenue :

$$h_d(n) = h_d(nT_e) = \frac{1}{F_e} \int_{-\frac{F_e}{2}}^{\frac{F_e}{2}} \overline{h_d}(f)\, e^{j2\pi f n T_e} df \tag{1.6}$$

$$h_d(t) = \frac{1}{F_e} \int_{-\frac{F_e}{2}}^{\frac{F_e}{2}} \overline{h_d}(f)\, e^{j2\pi f t} df \tag{1.7}$$

T_e et $F_e = \frac{1}{T_e}$ étant respectivement la période et la fréquence d'échantillonnage. Cette réponse impulsionnelle $h_d(n)$ étant généralement de support infini, la méthode de fenêtrage [Rabiner1975, pp. 88-104] consiste à effectuer une troncature de la séquence $h_d(n)$ en la multipliant par une fenêtre appropriée $w(n)$ de taille finie, dite fonction fenêtre, pour obtenir un filtre RIF dont la réponse impulsionnelle est donnée par :

$$h(n) = h_d(n).w(n) \tag{1.8}$$

La fenêtre intuitive que nous pouvons appliquer pour tronquer la réponse impulsionnelle est la fenêtre rectangulaire :

$$w(n) = \begin{cases} 1 & pour \quad n = 0, 1, ..., N \\ 0 & ailleurs \end{cases}$$

Ainsi, par une simple troncature des coefficients de $h_d(n)$, nous ne retenons que les $N+1$ premiers échantillons de $h_d(n)$ pour obtenir un filtre RIF $h(n)$ d'ordre N :

$$h(n) = \begin{cases} h_d(n) & pour \quad n = 0, 1, ..., N \\ 0 & ailleurs \end{cases}$$

Dans le domaine fréquentiel, cette opération de fenêtrage se traduit par une convolution du spectre $\overline{h_d}(f)$ du filtre désiré avec le spectre $W(f)$ de la fenêtre utilisée. Ainsi, le spectre du filtre synthétisé est donné par la relation :

$$\overline{h}(f) = \overline{h_d}(f) * W(f) \tag{1.9}$$

$\overline{h_d}(f)$ et $W(f)$ étant périodiques de période F_e, la convolution (1.9) s'écrit :

$$\overline{h}(f) = \frac{1}{F_e} \int_{-\frac{F_e}{2}}^{\frac{F_e}{2}} \overline{h_d}(\theta)\, W(f-\theta)\, d\theta \tag{1.10}$$

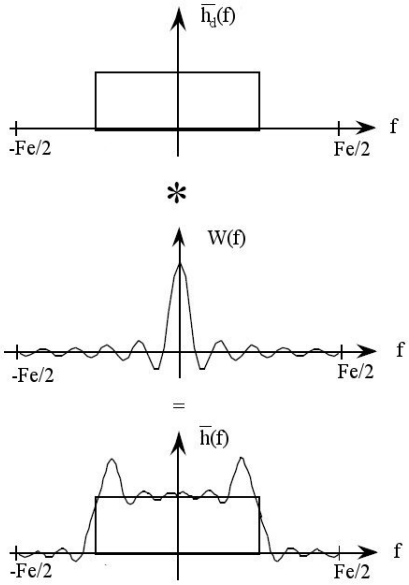

FIGURE 1.1 – Synthèse de filtre : méthode des fenêtres

La technique de fenêtrage entraîne donc une dégradation du spectre $\overline{h_d}(f)$ introduite par la convolution avec le spectre $W(f)$ de la fenêtre utilisée (Figure 1.1). Cette convolution fait apparaître des ondulations en bande passante et en bande atténuée et limite la raideur de coupure du filtre : le filtre obtenu possède une bande de transition non nulle. Pour minimiser ces ondulations et obtenir un filtre avec une bande de transition étroite, le spectre de la fenêtre utilisée doit avoir :

- un lobe principal étroit pour que la bande de transition le soit également,
- la plupart de l'énergie doit être concentrée dans le lobe principal afin d'obtenir un niveau réduit des lobes secondaires dans le spectre du filtre à obtenir.

Ces deux conditions constituent un dilemme. La première exige que le lobe principal soit étroit, alors que la seconde exige qu'il soit large. La fenêtre rectangulaire très simple à réaliser n'est pas forcément la mieux adaptée pour la synthèse des filtres. D'autres fenêtres (Hamming, Hanning, Blackman, Kaiser, etc.), ne présentant pas de discontinuités, ont été proposées pour mieux répondre à ce dilemme [Kaiser1966, Kaiser1974]. Sachant que l'atténuation des lobes secondaires par rapport au lobe principal correspond à celle de la bande affaiblie par rapport à la bande passante du filtre RIF à obtenir, et que la largeur du lobe principal est pratiquement égale à celle de la bande de

transition, un compromis entre ces deux facteurs doit être trouvé selon le tableau 1.1. Certaines fenêtres, comme celle de Kaiser, fournissent un paramètre β à régler permettant ainsi de maîtriser ce compromis.

TABLE 1.1 – Paramètres des fenêtres utilisées pour la conception des filtres RIF (A : atténuation maximale des lobes secondaires en décibels, Δ : largeur du lobe principal)

Fenêtres	A	Δ
Rectangulaire	$-13\,dB$	$2/N$
Triangulaire	$-24\,dB$	$4/N$
Hanning	$-32\,dB$	$4/N$
Hamming	$-42\,dB$	$4/N$
Blackman	$-57\,dB$	$6/N$
Kaiser ($\beta = 7.865$)	$-57\,dB$	$8/N$

Cette méthode de synthèse de filtre RIF présente l'avantage d'être simple à mettre en oeuvre. Cependant, elle aboutit à une synthèse sous-optimale des filtres puisqu'elle ne consiste pas à résoudre un problème d'optimisation bien défini. En outre, pour pouvoir calculer la réponse impulsionnelle désirée $h_d(n)$ par la transformée de Fourier inverse, il est nécessaire de connaître l'expression analytique de la réponse fréquentielle $h_d(f)$ du filtre désiré, ce qui n'est pas toujours le cas.

1.3.2 Méthode de l'échantillonnage en fréquence

Cette méthode consiste à calculer les coefficients de $h(n)$ par une simple transformée de Fourier discrète inverse [Gold1969, Rabiner1970]. Sa mise en oeuvre ne nécessite qu'un ensemble fini $\overline{h_d}(k)$ de K échantillons du spectre $\overline{h_d}(f)$. La séquence $\overline{h_d}(k)$ est définie par :

$$\overline{h_d}(k) = \overline{h_d}(f)_{f=k\frac{F_e}{K}} = \overline{h_d}\left(k\frac{F_e}{K}\right), \quad k = 0, 1, ..., K-1 \tag{1.11}$$

Les coefficients $h(n)$ sont alors donnés par la transformée de Fourier discrète inverse ($TFDI$) de $\overline{h_d}(k)$:

$$h(n) = TFDI\left(\overline{h_d}(k)\right) \tag{1.12}$$

soit :

$$h(n) = \frac{1}{K}\sum_{k=0}^{K-1} \overline{h_d}(k)\, e^{j2\pi nk/K} \quad n = 0, 1, ..., K-1 \tag{1.13}$$

Le spectre $\overline{h}(f)$ du filtre synthétisé est alors donné par la transformée de Fourier à temps discret de la réponse impulsionnelle $h(n)$ obtenue. Ce spectre $\overline{h}(f)$ peut être obtenu par une procédure

d'interpolation de la réponse en fréquence $\overline{h_d}(k)$ sachant que chaque échantillon de cette réponse pondère une fonction $A(f,k)$ donnée par :

$$A(f,k) = \frac{sin\left(\frac{K}{2}\left(f - k\frac{F_e}{K}\right)\right)}{sin\left(\frac{1}{2}\left(f - k\frac{F_e}{K}\right)\right)} \tag{1.14}$$

Ainsi le spectre du filtre obtenu est donné par :

$$\overline{h}(f) = \frac{1}{K}\sum_{k=0}^{K-1} A(f,k)\,\overline{h_d}(k) \tag{1.15}$$

Cette dernière équation montre bien que la méthode de synthèse des filtres RIF par la méthode d'échantillonnage en fréquence n'est pas vraiment une approximation, mais une approche d'interpolation du filtre $\overline{h}(f)$ à partir des K échantillons $\overline{h_d}(k)$ du filtre désiré. L'erreur entre $\overline{h}(f)$ et $\overline{h_d}(f)$ est nulle pour les fréquences $f = k\frac{F_e}{K}$, $k = 0, 1, ..., K-1$. Quant aux erreurs associées aux fréquences $f \neq k\frac{F_e}{K}$, elles sont finies et dépendent de K, lui même lié à l'ordre P du filtre RIF par la relation $K = P+1$. Cette méthode, rapide et simple, peut être utilisée pour la synthèse des filtres adaptatifs ou bien pour celle des filtres RIF dans le cas où ce procédé de synthèse ne constitue qu'une phase intermédiaire d'un algorithme complexe nécessitant une rapidité de calcul.

1.3.3 Méthodes d'optimisation

Méthode de Parks-McClellan

C'est la méthode optimale selon le critère de Chebyshev pour déterminer les coefficients d'un filtre RIF [McClellan1973]. Cet algorithme, n'est qu'une variante de l'algorithme de Remez appliqué à la synthèse des filtres RIF [Remez1934, Parks1972]. Cette méthode, basée sur une distribution uniforme de l'ondulation sur l'ensemble de la bande passante et sur une distribution uniforme de l'affaiblissement sur l'ensemble de la bande de coupure, recherche itérativement les coefficients $h(n)$ afin que pour une longueur minimale du filtre, le gabarit soit respecté au mieux. Les filtres RIF résultants, appelés aussi filtres minimax [Helms1971], possèdent une réponse en phase linéaire. Ils sont nettement plus performants que les filtres RIF fenêtrés et présentent par rapport à ces derniers les avantages suivants :

- Pour un gabarit identique, l'ordre d'un filtre RIF de Parks-McClellan est nettement inférieur.
- L'ondulation dans la bande passante et l'affaiblissement minimal dans la bande de coupure

sont configurables séparément.

Malgré ces avantages, cette méthode reste assez complexe à mettre en oeuvre en raison de son exigence en terme de coût de calcul.

Méthode des moindres carrés (Filtre least-square)

Cette méthode a pour objectif d'améliorer la qualité de l'approximation fournie par la méthode de l'échantillonnage en fréquence en considérant un nombre K d'échantillons qui soit supérieur à la taille du filtre RIF à obtenir, soit $K > P+1$, P étant l'ordre du filtre. Basée sur l'algorithme des moindres carrés, elle consiste à minimiser les carrées des écarts des ondulations entre la réponse en fréquence du filtre désiré et celle du filtre RIF à obtenir [Tufts1970, Kellogg1972, Lim1983].

L'obtention de la réponse impulsionnelle $h(n)$ d'ordre P consiste à résoudre le système K équations à $P+1$ inconnues :

$$\sum_{n=0}^{P} h(n) e^{-j2\pi nk/K} = \overline{h_d}(k), \qquad k = 0, 1, ..., K-1 \tag{1.16}$$

Dans la méthode d'échantillonnage en fréquence, l'ordre P est égal à $K-1$, ainsi le système obtenu est formé de K équations à K inconnues :

$$\sum_{n=0}^{K-1} h(n) e^{-j2\pi nk/K} = \overline{h_d}(k), \qquad k = 0, 1, ..., K-1$$

soit,

$$TFD(h) = \overline{h_d} \Longleftrightarrow h = TFDI\left(\overline{h_d}\right)$$

Dans le cas où le nombre d'équations K est supérieur au nombre d'inconnues $P+1$, le système est surdéterminé. Une solution exacte de l'équation (1.16) peut ne pas exister. Ainsi, la méthode des moindres carrés permet d'obtenir les coefficients $h(n)$ du filtre qui minimise l'erreur quadratique définie par :

$$E = \sum_{k=0}^{K-1} \left| \sum_{n=0}^{P} h(n) e^{-j2\pi nk/K} - \overline{h_d}(k) \right|^2$$

Ce problème est facile à résoudre dans le cas où les échantillons $\overline{h_d}(k)$ sont équidistants. En effet,

l'utilisation du théorème de Parseval permet d'écrire l'erreur E sous la forme :

$$E = \sum_{n=0}^{K-1} |h(n) - h_d(n)|^2 \tag{1.17}$$

Le filtre h étant d'ordre P, $h(n) = 0$ pour $n > P$, cette erreur E peut se décomposer comme suit :

$$E = \sum_{n=0}^{P} |h(n) - h_d(n)|^2 + \sum_{n=P+1}^{K-1} |h_d(n)|^2 \tag{1.18}$$

où le premier terme $\sum_{n=0}^{P} |h(n) - h_d(n)|^2$ représente une erreur dépendante du choix des valeurs de $h(n)$ et où le second terme $\sum_{n=N}^{K-1} |h_d(n)|^2$ représente une erreur résiduelle constante. Ainsi, nous pouvons déduire que, pour les valeurs de n entre 0 et P, les valeurs de $h(n)$ qui minimisent l'erreur E sont égales à $h_d(n)$.

Dans le cas où les échantillons $\overline{h_d}(k)$ ne sont pas équidistants, nous pouvons utiliser une des méthodes de résolution des systèmes sur-déterminés telle que la méthode de pseudo-inverse.

Comparé à l'algorithme de Parks-McClellan, les filtres obtenus par cette méthode des moindres carrés possèdent des ondulations moindres dans la bande passante et un affaiblissement minimal dans la bande de coupure plus accentué. En revanche, la bande de transition des filtres est plus large dans le cas de ces filtres.

1.4 Réalisation directe des filtres RIF

La réalisation concrète d'un filtre numérique peut se faire soit en logique câblée : assemblage d'organes logiques, tels que les portes logiques, les mémoires, etc ..., soit plus facilement en logique programmée : organisation autour d'un calculateur de traitement du signal (DSP) ou autour d'un microprocesseur standard.

Un filtre numérique est généralement constitué des éléments suivants :

- des organes de retard : ce sont des registres à décalage jouant le rôle de mémoires retardées, pilotés par une horloge,
- des opérateurs arithmétiques (additionneurs et multiplieurs),
- des registres fournissant les coefficients de pondération du filtre.

La Figure 1.2 représente la forme directe, dite aussi forme transversale ou forme non récursive, de

réalisation d'un filtre RIF d'ordre N dont la réponse impulsionnelle est donnée par :

$$h(n) = \begin{cases} b_n & pour \quad 0 \leq n \leq N \\ 0 & ailleurs \end{cases}$$

Cette structure montrée, dans laquelle une multiplication par z^{-1} associée à la transformée en z

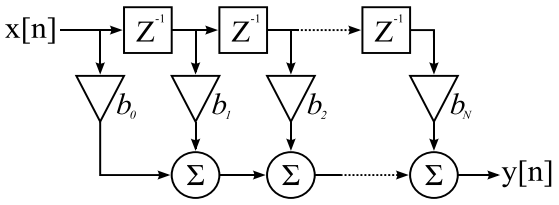

FIGURE 1.2 – Réalisation directe d'un filtre RIF

correspond à un registre à décalage, peut être implantée sur un DSP ou une carte FPGA .

Pour ce filtre d'ordre N, le calcul de chaque échantillon $y(n)$ de sortie selon la relation $y(n) = \sum_{r=0}^{N} b_r x(n-r)$ nécessite $N+1$ multiplications et N additions. Avant l'arrivée sur le marché des DSP, les filtres RIF étaient quasiment inutilisés étant donné que la puissance de calcul exigée est trop importante pour un filtre d'ordre élevé. Par exemple, pour un signal stéréo de 20 kHz échantillonné à 44.1 kHz, un filtrage RIF d'ordre 50 nécessite 50 multiplications et 51 additions entre deux échantillons successives de sortie, soit en moyenne une opération toutes les 226 ns. Bien que la structure MAC (multiply and accumulate) que nous retrouvons dans la famille de processeur orienté vers le filtrage permet d'exécuter en un seul cycle machine une opération d'addition et de multiplication, le coût et le temps d'exécution de cette réalisation directe de filtrage RIF reste considérable.

1.5 Réalisation indirecte des filtres RIF

1.5.1 Segmentation du signal

Avec le développement des systèmes des télécommunications, des applications de plus en plus gourmandes en charges de calcul et nécessitant le traitement de l'information en temps réel apparaissent. Souvent, une application de traitement numérique comporte une boucle à plusieurs étapes : décimation, filtrage, traitements numériques divers, interpolation, etc... Sachant qu'un traitement échantillon par échantillon représente une perte de performance énorme, il est souvent bien plus efficace de traiter un signal par blocs d'échantillons. Pour mettre en oeuvre ce traitement par blocs, des méthodes indirectes, dites méthodes de filtrage par blocs, ont été développées. Elles présentent

les avantages suivants :

- une réduction de la complexité de filtrage,
- un traitement en parallèle des différents blocs par des différents processeurs ou unités de calcul, ce qui réduira le temps global de traitement.

Deux méthodes de filtrage par blocs sont bien connues, la méthode overlap-save et la méthode overlap-add. Ces deux techniques de filtrage par blocs permettent la mise en oeuvre des filtres RIF par la transformée de Fourier discrète implémentée au moyen de l'algorithme de transformée de Fourier rapide.

1.5.2 Transformée de Fourier

Pour un signal continu $x(t)$, les transformées de Fourier directe et inverse sont définies par :

$$X(f) = \int x(t)\, e^{-j2\pi ft} dt \tag{1.19}$$

$$x(t) = \int X(f)\, e^{j2\pi ft} dt \tag{1.20}$$

Pour un signal numérique $x(k)$ de taille K, échantillonné à une période T_e, la transformée de Fourier à temps discret est définie par :

$$X(f) = \sum_{k=0}^{K-1} x(k)\, e^{-j2\pi fkT_e} \tag{1.21}$$

$X(f)$ étant périodique de période $F_e = \frac{1}{T_e}$, sa transformée inverse est définie par :

$$x(k) = x(kT_e) = \int_{-\frac{F_e}{2}}^{+\frac{F_e}{2}} X(f)\, e^{j2\pi fkT_e} df \tag{1.22}$$

La transformée de Fourier des signaux numériques étant continue, elle n'est donc pas sous une forme appropriée pour un traitement numérique. Vu l'importance particulière de cette transformation en traitement numérique des signaux, il est nécessaire de discrétiser la fréquence. En discrétisant f avec un pas d'échantillonnage $\triangle f = \frac{1}{K}$, les transformées de Fourier discrète (TFD) directe et inverse sont ainsi définies par :

$$X(n) = \sum_{k=0}^{K-1} x(k)\, e^{-j2\pi \frac{kn}{K}}, \quad n = 0, 1, ..., K-1 \tag{1.23}$$

$$x(k) = \frac{1}{K} \sum_{n=0}^{K-1} X(n) e^{j2\pi \frac{kn}{K}}, \quad k = 0, 1, ..., K-1 \tag{1.24}$$

La transformée de Fourier discrète a pris son importance depuis la découverte d'un algorithme rapide de calcul, appelé transformée de Fourier rapide (TFR).

1.5.3 Transformée de Fourier rapide

La Transformée de Fourier rapide (TFR) n'est autre qu'un algorithme de calcul rapide de la TFD. Depuis sa découverte en 1965 par Cooley et Tukey [Cooley1965], la TFR a révolutionné le traitement numérique des signaux. Son efficacité réside dans la réduction du nombre d'opérations nécessaires, en particulier le nombre des multiplications, et donc dans sa rapidité en terme de temps de calcul.

En effet, selon la formule donnée en (1.23) et (1.24), le calcul de la TFD d'un signal numérique de taille K requiert K^2 multiplications et un nombre d'additions du même ordre. L'idée principale de l'algorithme de Cooley-Tukey consiste à décomposer le calcul de la transformée de Fourier discrète d'une séquence en un calcul de deux transformées de tailles identiques égales à la moitié de la taille de la transformée initiale. Depuis que Cooley et Tukey ont démontré que leur calcul de TFD, basé sur leur structure de Butterfly, peut se faire avec une complexité de l'ordre de $Klog_2K$, plusieurs algorithmes ont été développés pour réduire davantage cette complexité. L'algorithme le plus utilisé est celui de 'split-radix FFT' [Malvar1992]. Ce dernier, très efficace, présente l'avantage de nécessiter moins de multiplications et d'additions que tous les autres algorithmes de calcul de la TFD.

Pour un signal d'entrée dont les valeurs appartiennent au corps des complexes $\mathbb{C}$, le nombre de multiplications réelles et le nombre d'additions réelles nécessaires pour le calcul d'une TFD à K points au moyen de l'algorithme 'split-radix FFT' sont respectivement donnés par :

$$N_{mul}(K) = K(log_2K - 3) + 4 \tag{1.25}$$

$$N_{add}(K) = 3K(log_2K - 1) + 4 \tag{1.26}$$

Pour un signal d'entrée dont les valeurs appartiennent au corps des réels $\mathbb{R}$, ces nombres sont respectivement donnés par :

$$N_{mul}(K) = \frac{K}{2}(log_2K - 3) + 2 \tag{1.27}$$

$$N_{add}(K) = \frac{K}{2}(3log_2K - 5) + 4 \tag{1.28}$$

Par la suite, nous nous baserons sur ces résultats pour déterminer la complexité des algorithmes de filtrage par blocs utilisant la TFR.

1.6 Filtrage par transformée de Fourier rapide

L'idée de base des deux méthodes de filtrage par blocs overlap-save et overlap-add consiste à calculer la convolution via la transformée de Fourier discrète. En effet, d'après les propriétés de la TFD, le produit de convolution circulaire de deux séquences, $\{x_n\}$ et $\{h_n\}$, de longueur M chacune, peut être obtenu indirectement via la transformée de Fourier discrète selon l'équation suivante :

$$\{y_n\} = \{x_n\} \circ \{h_n\} = TFDI\left(TFD\left(\{x_n\}\right) \bullet TFD\left(\{h_n\}\right)\right) \tag{1.29}$$

où $\circ$ et $\bullet$ désignent respectivement les opérations du produit de convolution circulaire et de multiplication terme à terme.

Ce calcul de produit de convolution nécessite :

- le calcul de deux transformées directes : $\{\overline{x_n}\} = TFD\left(\{x_n\}\right)$ et $\left\{\overline{h_n}\right\} = TFD\left(\{h_n\}\right)$,
- la multiplication terme à terme des séquences ainsi obtenues : $\{\overline{y_n}\} = \{\overline{x_n}\} \bullet \left\{\overline{h_n}\right\}$,
- le calcul de la transformée inverse de la séquence finale : $\{y_n\} = TFDI\left(\{\overline{y_n}\}\right)$.

L'intérêt de ce calcul indirect de convolution est la possibilité de calculer la TFD et la $TFDI$ au moyen de la TFR, permettant ainsi une large réduction du coût de calcul de la convolution circulaire, comparé à l'algorithme de calcul direct de cette convolution.

Cependant, les avantages de ce calcul ne peuvent pas être exploités directement dans le filtrage numérique. En effet, dans la plupart des applications de filtrage, le problème qui se pose est d'implanter la convolution linéaire de deux séquences plutôt que la convolution circulaire. Le traitement au moyen de la TFD d'un signal décomposé en blocs disjoints est entaché d'un défaut systématique lié aux discontinuités induites par le découpage. Les études faites pour pallier ce défaut ont abouti à deux types, devenus classiques, de mise en oeuvre des méthodes rapides de filtrage qui sont [Oppenheim1989] :

- la méthode d'overlap-save (recouvrement avec mémorisation des blocs d'entrée),
- la méthode d'overlap-add (recouvrement avec addition des blocs de sortie).

1.6.1 Convolution linéaire - convolution circulaire

Pour illustrer la différence entre une convolution linéaire et une convolution circulaire, nous prenons le cas d'un signal $x(n)$ de taille $N_x = 8$, $x\,(n) = [1, 2, ..., 8]$, et d'un filtre $h(n)$ d'ordre $P = 2$, $h(n) = [9, 10, 11]$. Pour la convolution linéaire, le signal résultant $y\,(n)$, de taille $N_y = N_x + P = 10$, est donné par :

$$y\,(n) = \sum_{m=0}^{2} h\,(m)\,x\,(n-m)\,, \quad n = 0, 1, ..., 9$$

$$\begin{cases} y\,(0) &= & 9*1 & = & 9 \\ y\,(1) &= & 9*2+10*1 & = & 28 \\ y\,(2) &= & 9*3+10*2+11*1 & = & 58 \\ y\,(3) &= & 9*4+10*3+11*2 & = & 88 \\ y\,(4) &= & 9*5+10*4+11*3 & = & 118 \\ y\,(5) &= & 9*6+10*5+11*4 & = & 148 \\ y\,(6) &= & 9*7+10*6+11*5 & = & 178 \\ y\,(7) &= & 9*8+10*7+11*6 & = & 208 \\ y\,(8) &= & 10*8+11*7 & = & 157 \\ y\,(9) &= & 11*8 & = & 88 \end{cases}$$

Une façon d'illustrer ce calcul est de considérer la fenêtre $f = [11, 10, 9]$, formée à partir des coefficients de la séquence h prise en sens inverse, et de la faire glisser le long du signal x pour obtenir le signal de sortie y comme montré dans la figure 1.3.

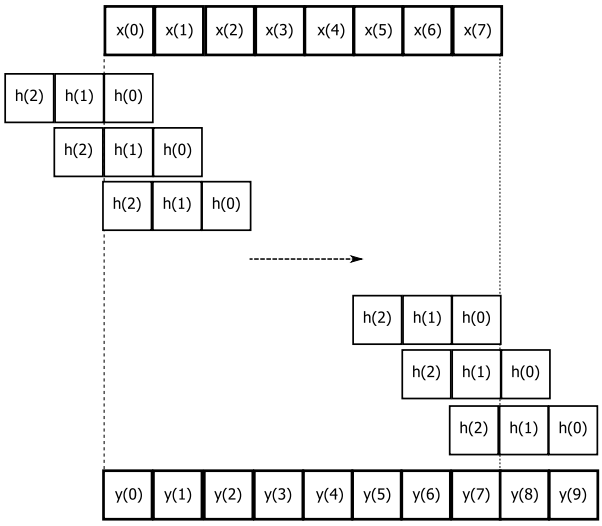

FIGURE 1.3 – Schéma de calcul de la convolution linéaire

En ajoutant des zéros à la suite de $h(n)$, pour obtenir la séquence h_c de taille 8, le signal $y_c(n)$ résultant de la convolution circulaire est de taille $M = 8$ et il est donné par :

$$y_c(n) = \sum_{m=0}^{7} h_c(m)\, x(\langle n-m \rangle_8)$$

où $\langle . \rangle_q$ désigne l'opérateur de calcul modulo q. Nous avons donc :

$$\begin{cases} y_c(0) = 9*1+10*8+11*7+0*6+0*5+0*4+0*3+0*2 = 166 \\ y_c(1) = 9*2+10*1+11*8+0 = 116 \\ y_c(2) = 9*3+10*2+11*1+0 = 58 \\ y_c(3) = 9*4+10*3+11*2+0 = 88 \\ y_c(4) = 9*5+10*4+11*3+0 = 118 \\ y_c(5) = 9*6+10*5+11*4+0 = 148 \\ y_c(6) = 9*7+10*6+11*5+0 = 178 \\ y_c(7) = 9*8+10*7+11*6+0 = 208 \end{cases}$$

Nous pouvons remarquer que l'on a $y(i) = y_c(i)$ pour $i = 2, 3, ..., 7$. Ce résultat peut être généralisé pour des tailles quelconques N_x de la séquence x et un ordre quelconque P du filtre h. Cette généralisation peut s'écrire :

$$y(i) = y_c(i) \quad pour \;\; i = P, P+1, ..., N_x - 1$$

1.6.2 Overlap-save (recouvrement des blocs d'entrée)

Pour obtenir une égalité entre la convolution linéaire et la convolution circulaire, la méthode overlap-save consiste à subdiviser le signal d'entrée x en blocs x_k de taille M présentant un recouvrement partiel de P échantillons. Le recouvrement des blocs permet de remédier aux effets non désirés qui se produisent sur les premiers P éléments de chaque bloc. A la fin du traitement de chaque bloc par la convolution circulaire, nous ne gardons que les $L = M - P$ échantillons à droite. Le principe de cette méthode est représenté par la figure 1.4.

Sachant que la convolution linéaire de deux séquences n'est pas modifiée si l'on complète les séquences par des zéros à gauche ou à droite, nous introduisons P zéros au début du premier bloc d'entrée et nous complétons si nécessaire par des zéros le dernier bloc pour avoir un bloc de taille

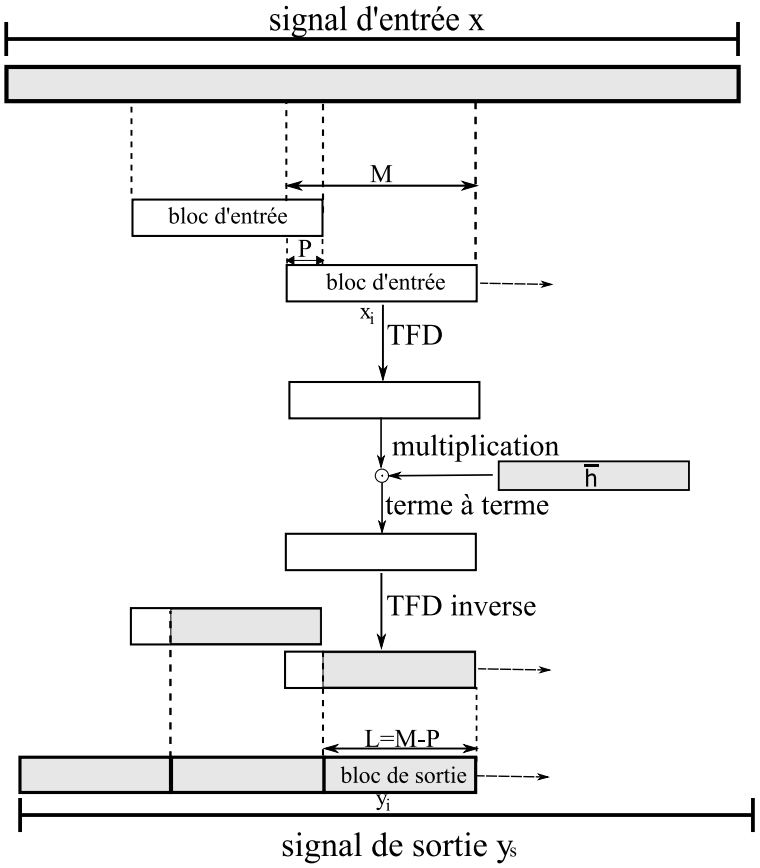

FIGURE 1.4 – Schéma d'overlap-save

M. Ainsi le premier bloc x_1 est donné par :

$$x_1(n) = \begin{cases} 0 & 0 \leq n \leq P-1 \\ x(n-P) & P \leq n \leq M-1 \end{cases}, \quad n = 0, 1, ..., M-1$$

Les autres blocs x_k sont tels que :

$$x_k(n) = x(n + (k-1)L - P), \quad n = 0, 1, ..., M-1, \quad k = 2, 3, ...$$

1.6.3 Overlap-add (addition des blocs de sortie)

Basé sur la propriété de la linéarité de la convolution linéaire, le principe de cette méthode, donnée en figure 1.5, consiste à décomposer le produit de convolution linéaire entre une séquence $x(n)$ et un filtre $h(n)$ en une somme de produits de convolution linéaire entre des séquences $x_k(n)$,

disjointes de taille L et issues de x, et le filtre $h(n)$ selon la relation :

$$y(n) = x(n) * h(n) = \sum_k x_k(n) * h(n) \tag{1.30}$$

Ensuite, le produit de convolution linéaire entre chaque séquence x_k, de taille L, et le filtre h, de

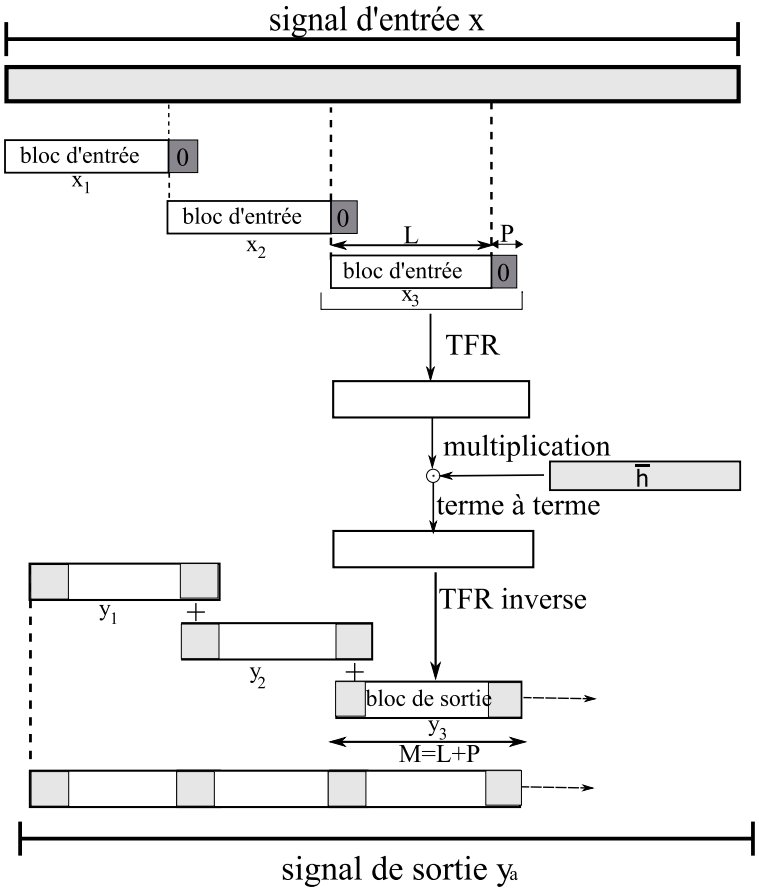

FIGURE 1.5 – Schéma d'overlap-add

taille $P+1$, est réalisé au moyen de le convolution circulaire des deux séquences $x_k^{'}$ et $h^{'}$, de même taille $L+P$, formées respectivement à partir des séquences x_k et h en complétant celles ci par des zéros : $x_k^{'} = [x_k, 0, ..., 0]$ et $h^{'} = [h, 0, ..., 0]$:

$$y_k = x_k * h = x_k^{'} \circ h^{'} \tag{1.31}$$

1.7 Coût du filtrage OLS-OLA

Bien que les processeurs DSP deviennent de plus en plus puissants, les multiplications restent plus complexes à réaliser par rapport aux opérations d'additions ou de décalage de bits. C'est pourquoi

nous considérons plus particulièrement les multiplications pour l'évaluation du coût des calculs des différents algorithmes. Nous supposons que la transformée utilisée est la transformée de Fourier discrète et que le coût de calcul de la transformée de Fourier rapide est celui donné par l'algorithme Split Radix FFT (SRFFT), considéré jusqu'à aujourd'hui le plus puissant pour le calcul de TFD. Selon l'équation (1.25), une TFD $M-$points d'une séquence de nombres complexes nécessite un nombre de multiplications réelles de l'ordre de $(Mlog_2M)$.

D'après les deux structures de filtrage OLS et OLA décrites auparavant, le traitement de chaque bloc nécessite :

- $\mathcal{O}(Mlog_2M)$ multiplications réelles pour le calcul de la transformée directe $M-$points.
- M multiplications complexes pour les multiplications terme à terme, soit $3M$ multiplications réelles.
- $\mathcal{O}(Mlog_2M)$ multiplications réelles pour le calcul de la transformée inverse $M-$points.

Au total, chaque bloc de L échantillons de sortie nécessite un nombre de multiplications réelles de l'ordre de $M(2log_2M+3)$. Ainsi, la complexité du filtrage, définie comme étant le nombre de multiplications réelles par échantillon de sortie est donnée par :

$$Comp = \frac{M(2log_2M+3)}{L} \tag{1.32}$$

1.8 Réalisation classique des méthodes OLS/OLA

Comme nous l'avons décrit précédemment dans le cas où les tailles L et M des blocs vérifient l'inégalité $L \leq M-P$, P étant l'ordre du filtre, le signal de sortie obtenu par les méthodes de filtrage par blocs OLS ou OLA utilisant la convolution circulaire est le même que celui obtenu par un filtrage direct utilisant la convolution linéaire. D'après l'équation (1.32), il est clair que, pour une valeur de M donnée, une diminution de L augmente la complexité du filtrage et inversement. Ainsi, en pratique, L est toujours choisi tel que $L = M-P$. Dans ce cas, la complexité de filtrage des méthodes OLS ou OLA est donnée par :

$$Comp = \frac{M(2log_2M+3)}{M-P} \tag{1.33}$$

Sachant que pour un filtre RIF d'ordre P, le filtrage direct nécessite $P+1$ multiplications complexes par échantillon, soit $3(P+1)$ multiplications réelles, nous pouvons en déduire que les deux

techniques OLS et OLA permettent une réduction de la complexité de calcul dans le cas où :

$$\frac{M\left(2log_2M+3\right)}{M-P}<3\left(P+1\right)$$

Pour illustrer cette réduction, nous considérons le cas typique où les blocs d'entrée sont de taille $M=512$. Les valeurs de P vérifiant la condition ci-dessus sont telles que $21<P<490$. Ceci signifie que les méthodes de filtrage par blocs OLS et OLA réduisent le coût de calcul du filtrage RIF dans le cas où l'ordre P du filtre h est compris entre 21 et 490.

Bien que la méthode de filtrage par la TFR est extrêmement performante, cette technique est trop peu utilisée, soit par ignorance, soit par crainte de devoir implémenter l'algorithme TFR alors que l'algorithme RIF est si simple et que nous le retrouvons implémenté sur presque tous les DSP.

Pour réaliser un filtrage selon les méthodes OLS et OLA, le choix des valeurs du filtre h ainsi que les valeurs de M et de L se fait selon le schéma suivant :

- Synthèse d'un filtre RIF d'ordre P selon l'une des méthodes de synthèse classique décrites auparavant.
- Choix de M : pour une efficacité maximale de calcul de la TFR, le choix de M est souvent limité aux valeurs égales à une puissance de 2. Parmi ces valeurs, nous choisissons celle qui minimise la complexité $Comp=\frac{M(2log_2M+3)}{M-P}$.

L'implantation du filtre sera alors réalisée en fonction de ces valeurs ainsi déterminées.

1.9 Contexte de notre étude

Sachant que la complexité de calcul, les erreurs de distorsion, les erreurs de quantification et les erreurs de calcul sont les principaux facteurs qui déterminent la qualité d'un filtrage numérique, l'objectif de notre étude est alors d'optimiser ces deux techniques de filtrage par blocs en réduisant la complexité de calcul et en minimisant les différentes erreurs. Pour bien comprendre cette étude, nous procédons comme suit :

A partir d'une réponse en fréquence donnée $\overline{h_d}$, nous désirons réaliser un filtrage fréquentiel RIF par blocs, par exemple overlap-save. Pour cela une synthèse du filtre réalisée selon l'une des méthodes classiques de synthèse permet d'obtenir la réponse impulsionnelle $h\left(n\right)$, $n=0,1,...,P$, P étant l'ordre du filtre. La taille optimale M des blocs d'entrée est calculée selon le principe décrit dans la section 1.8. Ainsi, sur le schéma de principe du filtrage overlap-save, donné par la figure 1.4,

$\overline{h}$ correspond à la TFD M-points du filtre h obtenu et la taille optimale L des blocs de sortie est donnée par $L = M - P$.

Concernant les erreurs de distorsion du filtrage, deux questions se posent :

1. En gardant la même structure d'OLS avec les mêmes tailles, M des blocs d'entrée et L des blocs de sortie, peut-on obtenir une erreur de distorsion inférieure à celle obtenue avec le filtre h ? Autrement dit, peut-on optimiser les coefficients de $\overline{h}$?

   Pour répondre à cette première question, nous proposerons dans le deuxième chapitre une approche de synthèse de filtre optimal (à distorsion minimale). Cette approche est basée sur un développement de la structure matricielle du filtrage par blocs. Notre étude ne se limite pas à la synthèse des filtres blocs optimaux mais elle concerne aussi leur synthèse par des algorithmes de faible complexité. En effet, certains applications en télécommunications nécessitent des changements fréquents de la bande passante des filtres utilisés. Pour permettre de modifier simplement et rapidement les bandes passantes de ces filtres optimaux, un algorithme de très faible complexité est développé pour leur synthèse en temps réel.

2. En gardant la même structure de traitement par blocs et les mêmes tailles des blocs d'entrée M et de sortie L, peut-on réduire davantage l'erreur de distorsion en modifiant la structure interne du traitement de chaque bloc ?

   Pour répondre à cette deuxième question, nous proposerons dans le chapitre 3 un modèle généralisé de filtrage par blocs dans lequel le filtre bloc est représenté par une matrice notée G et non plus par un vecteur $\overline{h}$. Les structures classiques d'OLS et d'OLA ne sont qu'un cas particulier du modèle généralisé dans le cas où la matrice G est diagonale. Cette généralisation permet de faire un compromis entre la distorsion du filtre d'un côté et le coût de filtrage par blocs de l'autre côté. Quatre méthodes sont ainsi développées pour la synthèse du filtre bloc généralisé optimal. L'utilisation de l'une ou l'autre de ces méthodes dépend du type de la transformée utilisée (unitaire, non unitaire) et de la pondération des fréquences (pondéré ou non pondéré).

3. Le fait que le filtrage par blocs OLS/OLA nécessite moins de calculs que le filtrage direct laisse penser que ce filtrage par blocs permet de réduire les erreurs de distorsion. Cependant, ce simple raisonnement naïf n'est pas correct. En effet, dans le filtrage par blocs, une erreur supplémentaire due à la quantification des termes cosinus et sinus associés aux coefficients de

la TFD est introduite. Les erreurs de quantification et d'arrondi des calculs peuvent être très importantes si l'implantation se fait sur un processeur à virgule fixe. Le choix d'un processeur à virgule fixe peut être favorisé du fait que son coût de calcul et sa consommation sont largement réduits par rapport à un processeur à virgule flottante. Pour éliminer ces erreurs, nous proposerons dans le chapitre 5 une implémentation d'OLS et d'OLA mettant en oeuvre la transformée en nombres de Fermat (FNT : Fermat Number Transform). La FNT est une transformée en nombres entiers (NTT : Number Theoretic Transform) particulière, définie sur un corps de Galois d'ordre égal à un nombre de Fermat. Cette transformée, connue depuis une trentaine d'année mais très peu utilisée en traitement du signal, présente de nombreux avantages par rapport à d'autres transformées plus populaires, comme la transformée de Fourier discrète, lorsqu'il s'agit d'une implémentation en virgule fixe. Elle permet d'un coté une large réduction du nombre de multiplications nécessaires à la réalisation du produit de convolution circulaire et de l'autre côté un calcul sans erreur d'arrondi. Le chapitre 4 présente le principe et les propriétés fondamentales des transformées en nombres entiers, en particulier la FNT. Sachant qu'il n'existe, à notre connaissance, dans la littérature aucun algorithme de synthèse des filtres blocs basés sur la FNT, un algorithme de très faible complexité pour la synthèse de ces filtres sera développé dans le chapitre 5.

1.10 Conclusion

Ce chapitre a rappelé les notions de base du filtrage numérique à réponse impulsionnelle finie (RIF). Nous avons décrit le principe des méthodes classiques de synthèse des filtres RIF et leur mise en oeuvre par la méthode directe de filtrage linéaire. Le principe et l'intérêt des méthodes de filtrage par blocs, overlap-save et overlap-add, mettant en oeuvre l'algorithme de filtrage rapide au moyen de la transformée de Fourier rapide associée à la transformée de Fourier discrète étaient ensuite exposés. Une étude comparative entre le filtrage direct et le filtrage par blocs portant sur la complexité a permis de démontrer l'intérêt de ce type de filtrage par blocs.

Le chapitre suivant sera consacré à l'étude et au développement de notre nouvel algorithme de très faible complexité pour une synthèse optimale des filtres blocs.

Chapitre 2

Conception optimale des filtres blocs

2.1 Introduction

Les méthodes indirectes de filtrage par blocs, overlap-save (OLS) et overlap-add (OLA), sont des méthodes qui répondent au besoin de filtrage rapide. Les filtres RIF implémentés au moyen de ces méthodes OLS et OLA sont obtenus par l'une des méthodes classiques de synthèse des filtres RIF sans tenir compte de la structure particulière de ces deux implémentations. D'autre part, certaines applications récentes en télécommunications nécessitent des changements fréquents des bandes passantes des filtres. Pour ces applications, le traitement en temps réel de l'information requiert non seulement des algorithmes rapides de filtrage mais aussi des méthodes rapides de synthèse des filtres pour permettre de modifier simplement et rapidement leurs bandes passantes. L'objectif de ce chapitre est d'optimiser le filtrage par blocs, en termes d'erreurs de distorsion, sans pour autant modifier le traitement à réaliser et de proposer des algorithmes rapides pour la synthèse de ces filtres blocs optimaux.

Dans ce chapitre, nous allons développer les modèles mathématiques propres aux structures de filtrage par blocs OLS et OLA. En nous basant sur ces modèles, nous proposons une nouvelle approche pour la synthèse des filtres blocs optimaux. L'optimisation est basée sur la minimisation de l'erreur quadratique entre le filtrage désiré et le filtrage par blocs permettant ainsi d'obtenir un filtrage à moindre distorsion. Des algorithmes rapides de synthèse des filtres blocs optimaux sont proposés. La performance de ces algorithmes est basée sur l'utilisation de la décomposition des matrices circulantes faisant appel à la transformée de Fourier discrète. Une comparaison entre le filtrage OLS et le filtrage OLA montre que leurs composantes de repliement de spectre sont

différentes.

Les résultats de ce chapitre ont fait l'objet de la publication de deux articles IEEE [Daher2008, Daher2009-a].

Bien que les processeurs DSP récents aient une puissance de calcul de plus en plus importante, les multiplications restent bien plus complexes à réaliser que les additions ou les décalage de bits. C'est pourquoi, nous nous intéressons au nombre de multiplications pour l'évaluation du coût de calcul des différents algorithmes.

2.2 Notations

La première ligne ou colonne d'une matrice sera d'indice 0. Les principales notations utilisées dans ce chapitre sont les suivantes :

F_N la matrice carrée d'ordre N associée à la TFD directe $N-$points,

F_N^{-1} la matrice inverse de F_N associée à la TFD inverse $N-$points,

$\mathbf{0}_{m\times n}$ la matrice nulle de taille $m \times n$,

$\boldsymbol{I}_m$ la matrice identité d'ordre m.

S la matrice sélection de taille $L \times M$. C'est une matrice binaire qui permet de sélectionner L éléments centraux d'un vecteur formé de M éléments. En notant $d = \frac{M-L}{2}$, on a :

$$S = [\mathbf{0}_{L\times d}\, \boldsymbol{I}_L\, \mathbf{0}_{L\times d}] \tag{2.1}$$

S^T la matrice transposée de S. C'est une matrice binaire qui permet de former un vecteur de taille M à partir d'un vecteur de taille $L = M - 2d$ en ajoutant $2d$ zéros, d zéros de chaque côté, au vecteur initial :

$$S^T = \begin{bmatrix} \mathbf{0}_{d\times L} \\ \boldsymbol{I}_L \\ \mathbf{0}_{d\times L} \end{bmatrix} \tag{2.2}$$

P_M la matrice permutation d'ordre M associée à l'opération de décalage cyclique à droite d'un élément,

$$P_M = \begin{bmatrix} \mathbf{0}_{1\times M-1} & \boldsymbol{I}_1 \\ \boldsymbol{I}_{M-1} & \mathbf{0}_{M-1\times 1} \end{bmatrix} \tag{2.3}$$

P_M^{-1} la matrice inverse de P_M associée à l'opération de décalage cyclique à gauche d'un élément,

$$P_M^{-1} = \begin{bmatrix} \mathbf{0}_{M-1\times 1} & \boldsymbol{I}_{M-1} \\ \boldsymbol{I}_1 & \mathbf{0}_{1\times M-1} \end{bmatrix} \tag{2.4}$$

$\langle . \rangle_q$ l'opération modulo q,

$|.|$ le module d'un nombre complexe.

Pour une matrice A de taille $m \times n$, les notations suivantes désignent :

A^T la matrice transposée de A,

$A^\star$ la matrice transposée conjuguée de A,

$A(:, j)$ le vecteur $m \times 1$ formé par les éléments de la $j^{ème}$ colonne de la matrice A,

$A(i, :)^T$ le vecteur $n \times 1$ formé par les éléments de la $i^{ème}$ ligne de la mtrice A,

$A(i, j)$ l'élément situé à l'intersection de la $i^{ème}$ ligne et de la $j^{ème}$ colonne de la matrice A.

2.3 Filtrage désiré

Propriété 2.1

La transformée de Fourier discrète K-points d'un signal x de taille LK, $L et K \in \mathbb{N}$, est égale à la somme des transformées de Fourier discrètes K-points des L signaux $x_l(n)$, $l = 0, 1, ..., L-1$, de taille K chacun tel que $x_l(n) = x(lK + n)$, $n = 0, 1, ..., K-1$.

Démonstration

Soit $x(n)$, $n = 0, 1, ..., LK-1$, un signal de taille $Z = LK$, $L \in \mathbb{N}$.

Les transformées de Fourier discrètes K-points des signaux $x_l(n)$ sont données par :

$$X_l(k) = \sum_{n=0}^{K-1} x_l(n) e^{-j2\pi \frac{nk}{K}}, \quad k = 0, 1, ..., K-1 \tag{2.5}$$

La transformée de Fourier discrète K-points du signal global x est donnée par :

$$X(k) = \sum_{n=0}^{LK-1} x(n) e^{-j2\pi \frac{nk}{K}}, \quad k = 0, 1, ..., K-1 \tag{2.6}$$

En développant cette dernière équation, nous obtenons :

$$\begin{array}{rcccccc} X(k) & = & \sum_{n=0}^{K-1} x(n) e^{-j2\pi \frac{nk}{K}} & + & \cdots & + & \sum_{n=(L-1)K}^{LK-1} x(n) e^{-j2\pi \frac{nk}{K}} \\ & = & \sum_{n=0}^{K-1} x_0(n) e^{-j2\pi \frac{nk}{K}} & + & \cdots & + & \sum_{n=0}^{K-1} x_{L-1}(n) e^{-j2\pi \frac{(n+(L-1)K)K}{K}} \\ & = & \sum_{n=0}^{K-1} x_0(n) e^{-j2\pi \frac{nk}{K}} & + & \cdots & + & e^{-j2\pi(L-1)k} \sum_{n=0}^{K-1} x_{L-1}(n) e^{-j2\pi \frac{nk}{K}} \\ & = & X_0(k) & + & \cdots & + & e^{-j2\pi(L-1)k} X_{L-1}(k) \end{array}$$

Sachant que, pour $l \in \{1, 2, ..., L-1\}$, $e^{-j2\pi lk} = \left(e^{-j2\pi}\right)^{lk} = (1)^{lk} = 1$, $X(k)$ s'écrit :

$$X(k) = X_0(k) + X_1(k) + ... + X_{L-1}(k) \tag{2.7}$$

La propriété est ainsi démontrée.

2.3.1 Structure matricielle du filtrage désiré

Le spectre d'un signal numérique, échantillonné à une fréquence d'échantillonnage F_e, est périodique de période F_e. Dans la pratique, les spectres des signaux numériques sont analysés avec une certaine résolution fréquentielle θ. Nous supposons, sans inconvénient, que θ est choisie de telle sorte que le rapport $\frac{F_e}{\theta}$ soit égal à un entier K, soit $K = \frac{F_e}{\theta}$. Ainsi, le choix de K est équivalent à celui de la résolution fréquentielle θ et inversement.

En se basant sur la propriété 2.1 démontrée ci-dessus, l'ensemble des signaux numériques de taille K et de période d'échantillonnage F_e et l'ensemble des réponses en fréquence de résolution $\theta = \frac{F_e}{K}$ sont reliés par une bijection donnée par la TFD $K-$points. Ainsi, sans perte de généralité,

nous pouvons considérer par la suite que les signaux d'entrée sont de taille K.

Nous représenterons à présent les signaux et leurs TFD par des vecteurs. Pour un signal d'entrée x, la TFD $\overline{y_d}$ du signal de sortie désiré y_d est donnée par :

$$\overline{y_d} = \overline{\overline{H_d}}\overline{x} \tag{2.8}$$

$\overline{x}$ étant la TFD du signal d'entrée x, $\overline{x} = F_K x$, $\overline{\overline{H_d}}$ est la matrice diagonale de taille K telle que :

$$\overline{\overline{H_d}}(k,k) = f(k), \quad k = 0, 1, ..., K-1 \tag{2.9}$$

$f(k)$, $k : 0, 1, ..., K-1$, est la réponse en fréquence du filtre désiré.

Le signal de sortie désiré y_d est donné par la TFD inverse de $\overline{y_d}$, soit d'après l'équation (2.8) :

$$y_d = F_K^{-1}\overline{\overline{H_d}}\overline{x} \tag{2.10}$$

Sachant que $I_K = F_K F_K^{-1}$, I_k étant la matrice identité, y_d peut s'écrire :

$$y_d = F_K^{-1}\overline{\overline{H_d}}F_K F_K^{-1}\overline{x} \tag{2.11}$$

soit :

$$y_d = H_d x \tag{2.12}$$

où H_d est la matrice carrée de taille K définie par :

$$H_d = F_K^{-1}\overline{\overline{H_d}}F_K \tag{2.13}$$

2.4 Retard des filtres RIF

Dans la synthèse des filtres RIF, les filtres causaux d'ordre P, $h(n)$, $n = 0, 1, ..., P$, sont généralement obtenus à partir des filtres non causaux $h'(n)$ de phase nulle, en introduisant un retard temporel selon :

$$h'(n) = h\left(n - \left[\frac{P}{2}\right]\right), \quad n = 0, 1, ..., P \tag{2.14}$$

$\left[\frac{P}{2}\right]$ correspond à la partie entière de $\frac{P}{2}$. D'après les propriétés de la transformée en Fourier, l'effet de ce retard se limite à une introduction d'un déphasage linéaire en fréquence. Pour illustrer l'effet de ce retard sur le signal de sortie, nous considérons un signal d'entrée $x(n) = [x_0, x_1, ..., x_7]$ de taille $N = 8$, et un filtre RIF causal d'ordre $P = 4$ donné par $h'(n) = [h'_0, h'_1, h'_2, h'_3, h'_4]$. Le signal $y'(n)$ de sortie du filtre RIF causal est donné par :

$$y'(n) = \sum_{m=0}^{4} h'(m)\, x(n-m) \quad n = 0, 1, ..., 11 \tag{2.15}$$

$$\begin{cases} y'(0) = h'(0)\, x(0) \\ y'(1) = h'(0)\, x(1) + h'(1)\, x(0) \\ y'(2) = h'(0)\, x(2) + h'(1)\, x(1) + h'(2)\, x(0) \\ y'(3) = h'(0)\, x(3) + h'(1)\, x(2) + h'(2)\, x(1) + h'(3)\, x(0) \\ y_{'}(4) = h'(0)\, x(4) + h'(1)\, x(3) + h'(2)\, x(2) + h'(3)\, x(1) + h'(4)\, x(0) \\ \vdots \quad \vdots \qquad\qquad \vdots \end{cases}$$

Le filtre non causal correspondant est donné par :

$$h(n) = h'(n+2), \;\; n = -2, -1, 0, 1, 2 \tag{2.16}$$

Le signal de sortie correspondant au filtre non causal est donné par :

$$y(n) = \sum_{m=-2}^{2} h(m)\, x(n-m) \quad n = -2, -1, ..., 9 \tag{2.17}$$

$$\begin{cases} y(-2) = h(-2)\, x(0) = y'(0) \\ y(-1) = h(-2)\, x(1) + h(-1)\, x(0) = y'(1) \\ y(0) = h(-2)\, x(2) + h(-1)\, x(1) + h(0)\, x(0) = y'(2) \\ y(1) = h(-2)\, x(3) + h(-1)\, x(2) + h(0)\, x(1) + h(1)\, x(0) = y'(3) \\ y(2) = h(-2)\, x(4) + h(-1)\, x(3) + h(0)\, x(2) + h(1)\, x(1) + h(2)\, x(0) = y'(4) \\ \vdots \quad \vdots \qquad\qquad\qquad\qquad\qquad \vdots \quad \vdots \end{cases}$$

De ces équations, nous en déduisons la relation suivante :

$$y'(n) = y(n-2), \;\; n = 0, 1, ..., 9 \tag{2.18}$$

Dans le cas général d'un signal x de taille K, traité par un filtre d'ordre P, le signal de sortie $y(n)$ obtenu à l'issu du filtrage non causal et le signal de sortie $y'(n)$ obtenu à partir du filtrage causal sont liés par la relation :

$$y'(n) = y\left(n - \left[\frac{P}{2}\right]\right), \quad n = 0, 1, ..., K + P \tag{2.19}$$

Un retard de $\left[\frac{P}{2}\right]$ échantillons est alors obtenu à la sortie. Ceci correspond à un retard temporel de $\frac{\left[\frac{P}{2}\right]}{F_e}$ secondes, F_e étant la fréquence d'échantillonnage. A titre d'exemple, un filtre RIF d'ordre $P = 20$ appliqué à un signal numérique échantillonné à une fréquence de $1\,kHz$ introduit un retard de $10\,ms$.

2.5 Équivalence entre la convolution linéaire et la convolution circulaire

L'équivalence, connue dans la littérature et montrée dans le chapitre 1, entre la convolution linéaire et la convolution circulaire est développée pour le cas des filtres RIF causaux. Bien que l'implémentation directe des filtres RIF exige que les filtres soient causaux, cette condition n'est pas nécessaire dans l'implémentation avec les techniques OLS et OLA. Ainsi, pour supprimer le retard de filtre, nous visons dans notre étude à implémenter les filtres RIF non causaux. Pour cela, nous développerons ci-dessous deux propriétés, propriété 2.2 et propriété 2.3, qui sont basées sur les propriétés de la périodicité de la convolution circulaire et seront utilisées respectivement pour développer nos approches OLS et OLA non causals.

Notons que, un produit de convolution linéaire entre deux séquences $x(n)$ et $h(n)$ de tailles respectives L_1 et L_2 telles que $x(n)$ est définie pour n allant de a à $a + L_1 - 1$ et $h(n)$ est définie pour n allant de b à $b + L_2 - 1$, est une séquence $y(n)$ de taille $L_1 + L_2 - 1$ définie pour n allant de $a + b$ à $a + b + L_1 + L_2 - 2$.

Propriété 2.2

Soient $x(n)$ et $h(n)$ deux séquences de tailles respectives M et $P + 1$. La séquence $x(n)$ est définie pour $n = 0, 1, ..., M - 1$ et la séquence $h(n)$ est définie pour $n = -\left[\frac{P}{2}\right], ..., \left[\frac{P+1}{2}\right]$. Nous

définissons les séquences $x_c(n)$ et $h_c(n)$ de tailles égales à M, $n = 0, 1, ..., M-1$ telles que :

$$x_c(n) = x(n)$$

$$h_c(n) = \left[h(0), ..., h\left(\left[\frac{P+1}{2}\right]\right), \mathbf{0}_{1\times M-P-1}, h\left(-\left[\frac{P}{2}\right]\right), ..., h(-1)\right]$$

En développant le calcul du produit de convolution linéaire $\{y(n)\} = \{x(n)\} * \{h(n)\}$, défini pour $n = -\left[\frac{P}{2}\right], ..., M-1+\left[\frac{P+1}{2}\right]$, et celui du produit de convolution circulaire $\{y_c(n)\} = \{x_c(n)\} \circ \{h_c(n)\}$, défini pour $n = 0, 1, ..., M-1$, nous obtenons :

$$y(i) = y_c(i) \quad pour \;\; i = \left[\frac{P}{2}\right], ..., M-1-\left[\frac{P+1}{2}\right] \tag{2.20}$$

N.B. $\forall\, P \in \mathbb{N}$, $\left[\frac{P+1}{2}\right] + \left[\frac{P}{2}\right] = P$.

Propriété 2.3

Soient $x(n)$ et $h(n)$ deux séquences de tailles respectives L et $P+1$. La séquence $x(n)$ est définie pour $n = \left[\frac{P+1}{2}\right], , ..., L-1+\left[\frac{P+1}{2}\right]$ et la séquence $h(n)$ est définie pour $n = -\left[\frac{P}{2}\right], ..., \left[\frac{P+1}{2}\right]$. Nous définissons les séquences $x_c(n)$ et $h_c(n)$ de tailles égales à $M = L+P$, $n = 0, 1, ..., M-1$ par :

$$x_c(n) = \left[\mathbf{0}_{1\times\left[\frac{P}{2}\right]}, x(n), \mathbf{0}_{1\times\left[\frac{P+1}{2}\right]}\right]$$

$$h_c(n) = \left[h(0), ..., h\left(\left[\frac{P+1}{2}\right]\right), \mathbf{0}_{1\times M-P-1}, h\left(-\left[\frac{P}{2}\right]\right), ..., h(-1)\right]$$

En développant le calcul du produit de convolution linéaire $\{y(n)\} = \{x(n)\} * \{h(n)\}$, défini pour $n = 0, ..., M-1$ et celui du produit de convolution circulaire $\{y_c\} = \{x_c\} \circ \{h_c\}$, défini également pour $n = 0, 1, ..., M-1$, nous pouvons déduire que :

$$y(i) = y_c(i) \quad pour \;\; i = 0, ..., M-1 \tag{2.21}$$

2.5.1 Implémentation du filtrage par blocs non causal

Dans le filtrage par blocs, le produit de convolution circulaire $\{y_c(n)\} = \{x_c(n)\} \circ \{h_c(n)\}$ sera réalisé au moyen de la transformée de Fourier discrète $y_c(n) = TFDI\left(TFD\left(\{x_c(n)\}\right) \bullet TFD\left(\{h_c(n)\}\right)\right)$. Sachant que le coefficients du filtre h_c sont fixes, le vecteur M-points $TFD\left(\{h_c(n)\}\right)$ sera calculé

une seule fois et stocké dans un registre.

Pour simplifier la compréhension de l'étude par la suite, nous supposons dans notre cas que P est un nombre pair égal à $2d$ ($\left[\frac{P+1}{2}\right] = \left[\frac{P}{2}\right]$).

2.5.2 Principe d'overlap-save non causal

Le principe de filtrage OLS non causal consiste à subdiviser le signal d'entrée x en blocs x_k de taille M présentant un recouvrement partiel de P échantillons. Après le traitement de chaque bloc $x_{k,c} = x_k$, nous ne retenons que les $L = M - P$ échantillons centraux du bloc $y_{k,c}$ afin d'obtenir les échantillons vérifiant la propriété 2.2 concernant l'égalité entre la convolution linéaire et la convolution circulaire. Le signal global de sortie y_s est alors obtenu par concaténation des différents blocs de sortie y_k. Ce principe d'overlap-save, illustré sur la figure 2.1, peut être décrit mathématiquement selon les étapes suivantes :

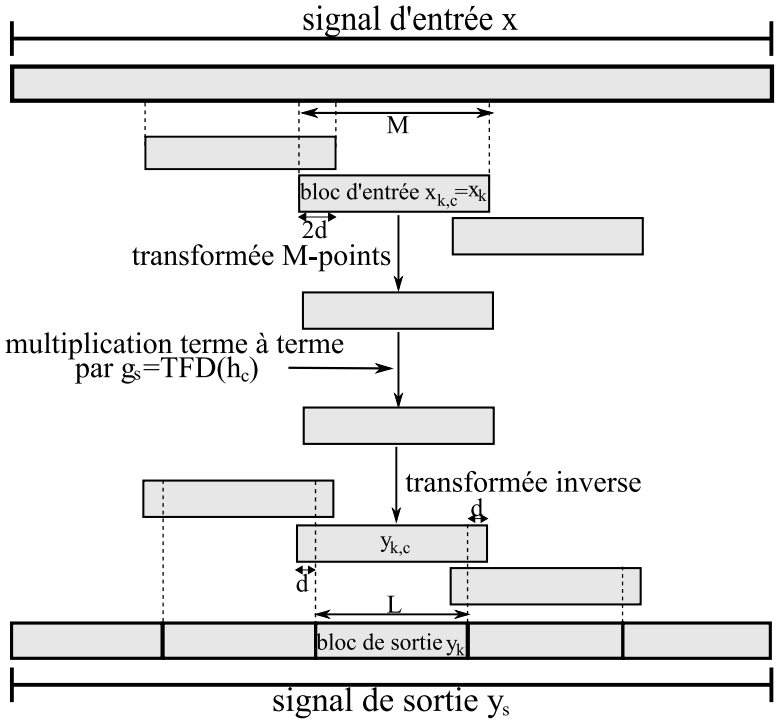

FIGURE 2.1 – Schéma d'overlap-save sans retard

1. Calcul de la transformée de Fourier discrète $M-$points du bloc d'entrée x_k, soit :

$$F_M x_k$$

2. Multiplication terme à terme du bloc obtenu par le vecteur $g_s = TFD\left(\{h_c(n)\}\right)$ de taille M,

soit :

$$g_s \bullet (F_M x_k)$$

3. Calcul de la transformée de Fourier discrète inverse $M-$points du bloc obtenu, soit :

$$y_{k,c} = F_M^{-1} (g_s \bullet (F_M x_k))$$

4. Élimination de d échantillons de chaque côté du bloc obtenu pour ne garder que les $L = M-2d$ échantillons centraux de ce bloc. Le bloc de sortie y_k ainsi obtenu est donné par :

$$y_k = S y_{k,c} = S F_M^{-1} (g_s \bullet (F_M x_k)) \tag{2.22}$$

où S désigne la matrice $L \times M$ permettant de sélectionner les L éléments centraux d'un vecteur formé de M éléments.

Sachant qu'une multiplication terme à terme entre deux vecteurs u et v correspond à la multiplication du vecteur v par la matrice diagonale U, telle que $diag\,(U) = u$, soit $u \bullet v = Uv$, la relation (2.22) s'écrit :

$$y_k = S F_M^{-1} G_s F_M x_k \tag{2.23}$$

G_s étant la matrice diagonale telle que $diag\,(G_s) = g_s$. Soit,

$$y_k = A_s x_k \tag{2.24}$$

où A_s est la matrice de taille $L \times M$ donnée par :

$$A_s = S F_M^{-1} G_s F_M \tag{2.25}$$

Sans perte de généralité, supposons que $K = bL$, avec b un entier. Le signal global de sortie y_s, de taille K, exprimé en fonction du signal d'entrée x, est donné par :

$$y_s = H_s x \tag{2.26}$$

H_s est la matrice de taille $K \times K$ contenant b copies de la matrice A_s, construite selon la structure donnée en figure 2.2.

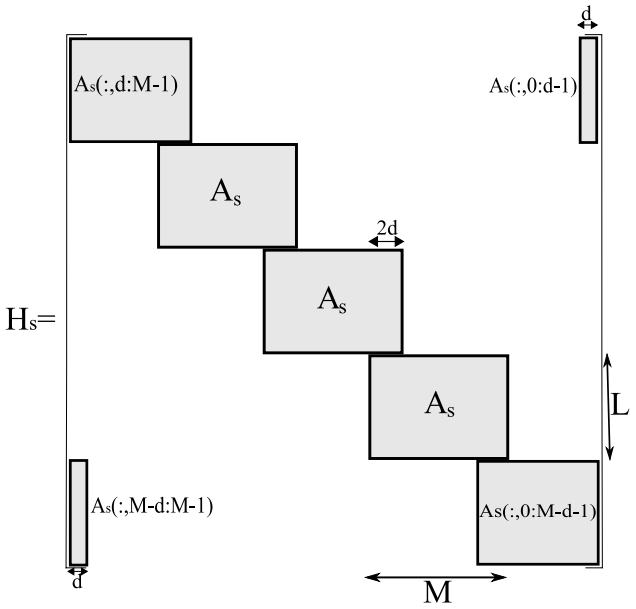

FIGURE 2.2 – Structure de la matrice H_s (cas de $b = 5$)

N.B. Le schéma est obtenu en supposant que, pour former le premier bloc d'entrée x_k, ses d premiers échantillons sont les d derniers échantillons du signal x, et que pour former le dernier bloc d'entrée, ses d derniers échantillons sont les d premiers échantillons du signal x. En pratique, ces échantillons sont pris égaux à zéro, ce qui introduit une erreur supplémentaire. Sachant que $d \ll K$, cette erreur pourra être négligée par la suite de notre étude.

Ainsi, la TFD $\overline{y_s}$ du signal de sortie y_s est donnée par :

$$\overline{y_s} = F_K y_s = \overline{\overline{H_s}}\,\overline{x} \tag{2.27}$$

$\overline{\overline{H_s}}$ est la matrice de taille $K \times K$ donnée par :

$$\overline{\overline{H_s}} = F_K H_s F_K^{-1} \tag{2.28}$$

2.5.3 Principe d'overlap-add non causal

Le principe d'overlap-add consiste à réaliser artificiellement une convolution linéaire au moyen d'un calcul de convolution circulaire. Le signal d'entrée est ainsi subdivisé en blocs disjoints x_k de taille L. Chaque bloc x_k est complété par $P = 2d$ zéros, d zéros de chaque côté, pour former un bloc de taille $M = L + P$ noté $x_{k,c}$.

D'après la propriété 2.3, le traitement par convolution circulaire de chaque bloc $x_{k,c}$ fournit un bloc $y_{k,c}$ identique à celui obtenu par un filtrage linéaire de x_k. Selon la propriété de linéarité de la convolution linéaire, le signal de sortie global y_a n'est autre que la somme des différents blocs $y_{k,c}$ obtenus. Ce principe, illustré sur la figure 2.3, peut être décrit mathématiquement selon les étapes suivantes :

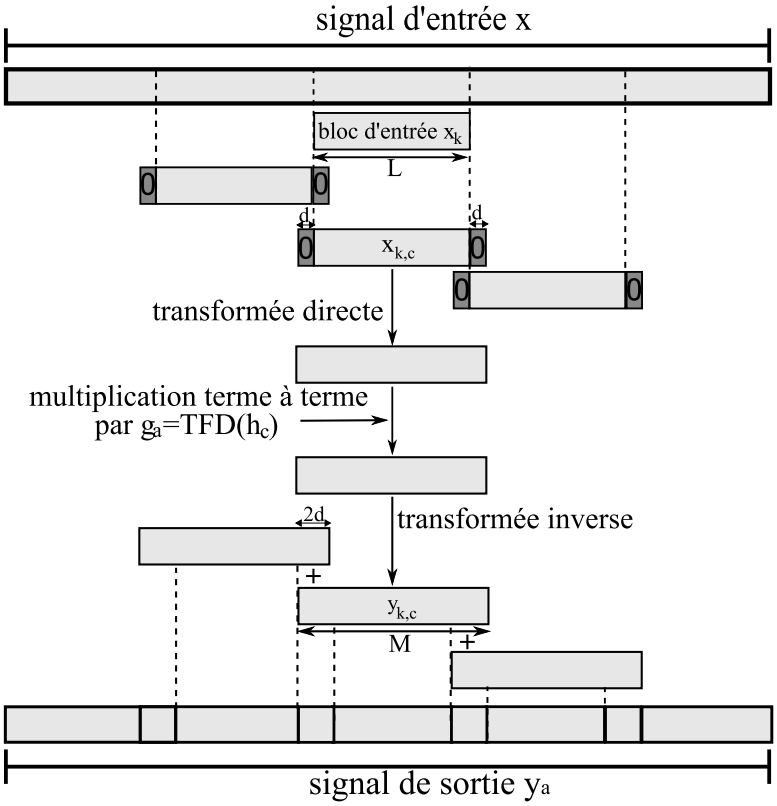

FIGURE 2.3 – Schéma overlap-add sans retard

1. Ajout de d zéros de chaque côté du bloc d'entrée x_k . Le bloc ainsi obtenu $x_{k,c}$, formé de $M = L + 2d$ échantillons est donné par :

$$x_{k,c} = \begin{bmatrix} 0_{d\times 1} \\ x_k \\ 0_{d\times 1} \end{bmatrix} = S^T x_k$$

où S^T est la matrice $M \times L$ transposée de la matrice sélection S.

2. Calcul de la transformée de Fourier discrète $M-$points du bloc ainsi formé :

$$F_M x_{k,c} = F_M S^T x_k$$

3. Multiplication terme à terme par le vecteur $g_a = TFD(\{h_c\})$ de taille M, soit :

$$g_a \bullet \left(F_M S^T x_k\right)$$

4. Calcul de la transformée de Fourier discrète inverse $M-$points pour obtenir le bloc de sortie $y_{k,c}$ donné par :

$$y_{k,c} = F_M^{-1}\left(g_a \bullet \left(F_M S^T x_k\right)\right) \tag{2.29}$$

En introduisant la matrice diagonale G_a telle que $diag(G_a) = g_a$, la relation (2.29) s'écrit :

$$y_{k,c} = F_M^{-1} G_a F_M S^T x_k \tag{2.30}$$

soit,

$$y_{k,c} = A_a x_k \tag{2.31}$$

où A_a est la matrice de taille $M \times L$ donnée par :

$$A_a = F_M^{-1} G_a F_M S^T \tag{2.32}$$

Sans perte de généralité, supposons que $K = bL$, avec b un entier. Le signal de sortie y_a, de taille K, obtenu au moyen du filtrage par blocs OLA est ainsi donné par :

$$y_a = H_a x \tag{2.33}$$

H_a est la matrice de taille $K \times K$ contenant b copies de la matrice A_a comme montré sur la figure 2.4.

N.B. Le schéma est obtenu en supposant que les d premiers et les d derniers échantillons du premier et du dernier blocs de sortie sont ajoutés respectivement aux d derniers échantillons et aux d premiers échantillons du signal de sortie y_a. En pratique, ces $2d$ échantillons sont supprimés sans être ajoutés ailleurs, ce qui introduit une erreur supplémentaire. Sachant que $d \ll K$, cette erreur pourra être négligée par la suite de notre étude.

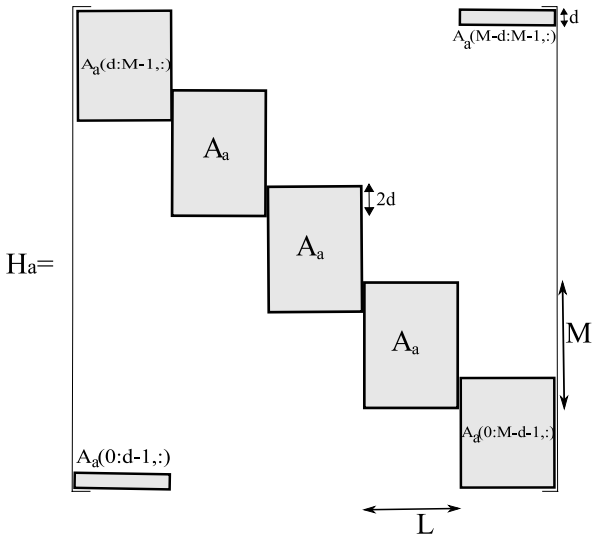

FIGURE 2.4 – Structure de la matrice H_a (cas de $b = 5$)

Ainsi, la réponse en fréquence $\overline{y_a}$ du signal de sortie est donnée par :

$$\overline{y_a} = F_K y_a = \overline{\overline{H_a}}\,\overline{x} \tag{2.34}$$

$\overline{\overline{H_a}}$ est la matrice de taille $K \times K$ donnée par :

$$\overline{\overline{H_a}} = F_K H_a F_K^{-1} \tag{2.35}$$

2.6 Critère d'optimisation des filtres blocs

Pour la suite, les notations en indice s et a respectivement relatives aux implémentations overlap-save et overlap-add sont omises sauf indication contraire. Par exemple, $\overline{y}$ désigne $\overline{y_s}$ dans le cas d'overlap-save et $\overline{y_a}$ dans le cas d'overlap-add.

La qualité d'un filtre numérique est évaluée par la mesure d'une certaine distance définie entre le spectre de sortie obtenu par le filtrage en blocs et celui de la sortie désirée. Dans notre cas, nous utilisons le critère standard de l'erreur quadratique moyenne (EQM). L'erreur à minimiser est ainsi donnée par :

$$e_{QM} = \boldsymbol{E}\left\{\sum_{k=0}^{K-1} \left|\overline{y}\,(k) - \overline{y_d}\,(k)\right|^2\right\} = \boldsymbol{E}\left\{\left\|\overline{y} - \overline{y_d}\right\|^2\right\} \tag{2.36}$$

où $\boldsymbol{E}$ désigne l'espérance mathématique, $\|.\|$ la norme euclidienne d'un vecteur et $|.|$ le module d'un nombre complexe. En remplaçant $\overline{y}$ et $\overline{y_d}$ par leurs expressions, données respectivement par (2.27)

ou (2.34) et (2.8), l'équation (2.36) s'écrit :

$$e_{QM} = \boldsymbol{E}\left\{\left\|\overline{\overline{H}}\overline{x} - \overline{\overline{H_d}}\overline{x}\right\|^2\right\} \tag{2.37}$$

L'optimisation du filtrage par blocs consiste alors à minimiser cette erreur quadratique. Dans notre étude, nous considérons le cas où les tailles des blocs M et L sont fixes. Ainsi, l'optimisation porte sur les éléments diagonaux de la matrice G. En utilisant l'hypothèse habituelle de considérer qu'un signal d'entrée x est un bruit blanc, donc de spectre fréquentiel $\overline{x}$ à répartition uniforme, minimiser e_{QM} revient à minimiser l'erreur e donnée par :

$$e = \left\|\overline{\overline{H}} - \overline{\overline{H_d}}\right\|_f^2 \tag{2.38}$$

où $\|.\|_f$ désigne la norme de Frobenius d'une matrice. En remplaçant $\overline{\overline{H}}$ et $\overline{\overline{H_d}}$ par leurs expressions données respectivement par $\overline{\overline{H}} = F_K H F_K^{-1}$ et $\overline{\overline{H_d}} = F_K H_d F_K^{-1}$, nous obtenons :

$$e = \left\|F_K H F_K^{-1} - F_K H_d F_K^{-1}\right\|_f^2 \tag{2.39}$$

Les matrices F_K et F_K^{-1} étant unitaires, elles conservent donc la norme de Frobenius. L'erreur e peut alors s'écrire :

$$e = \|H - H_d\|_f^2 \tag{2.40}$$

2.7 Les erreurs de distorsion

2.7.1 Filtre linéaire périodique variant dans le temps (LPVT)

Si certains travaux traitent explicitement des filtres linéaires périodiques variant dans le temps (LPVT, en anglais LPTV : Linear Periodical Time Varying), beaucoup d'autres utilisent des filtres LPVT de façon implicite. Ces filtres LPVT appartiennent à une classe plus générale de filtres que sont les filtres linéaires variant dans le temps (LVT, en anglais LTV : Linear Time Varying) et s'y distinguent par des propriétés de périodicité [Loeffler1984, Lernon1999, Chauvet2004]. Un filtre LTV n'a pas le même comportement que celui d'un filtre linéaire invariant dans le temps (LIT, en anglais LTI : Linear Time-Invariant). En effet, pour un filtre LIT, les signaux de sortie $y(n)$ et d'entrée $x(n)$ sont liés par la relation $y(n) = \sum_k h(k)\, x(n-k)$ où $h(k)$ est la réponse impulsionnelle du filtre.

Ceci se traduit dans le domaine fréquentiel par le fait que la composante d'indice k de la TFD du signal de sortie dépend uniquement de la composante de même indice de la TFD du signal d'entrée selon la relation $\overline{y}(k) = \overline{h}(k)\,\overline{x}(k)$ où $\overline{h}(k)$ est la réponse en fréquence du filtre. Cependant, pour un filtre LVT, les coefficients du filtre varient en fonction du temps. Ainsi, les signaux de sortie $y(n)$ et d'entrée $x(n)$ sont liés par la relation $y(n) = \sum_m H(n,m)\,x(n-m)$ où $H(n,m)$ désigne la réponse impulsionnelle du filtre à l'instant $t = nT_e$, T_e étant la période d'échantillonnage. En fréquentiel, ceci se traduit par le fait que la composante d'indice k du spectre $\overline{y}$ du signal de sortie ne dépend pas seulement de la composante de même indice du spectre $\overline{x}$ du signal d'entrée mais aussi de toutes les autres composantes selon la relation $\overline{y}(k) = \sum_l \overline{\overline{H}}(k,l)\,\overline{x}(l)$, où $\overline{\overline{H}}(k,l)$ est la transformée de Fourier bi-dimensionnelle de la matrice $H(n,m)$. Dans le cas où $H(n,m) = H(n+n_0,m)$ quels que soit n et m, le filtrage est de type LPVT de période n_0.

2.7.2 Les erreurs

Sans aucune contrainte sur les filtres blocs synthétisés, le filtrage par blocs OLS ou OLA est de type LPVT de période L. En effet, dans le cas du filtrage OLS par exemple, la structure de la matrice H_s donnée par la figure (2.2) permet d'écrire le signal de sortie $y_s(n)$ en fonction du signal d'entrée x selon la forme :

$$y_s(n) = \sum_{i=0}^{M-1} A_s\left(\langle n\rangle_L, i\right) x(n-d+i), \quad n = 0,1,...,K-1$$

En fréquentiel, l'équation $\overline{y} = \overline{\overline{H}}\overline{x}$ permet d'écrire :

$$\overline{y}(k) = \overline{\overline{H}}(k,k)\,\overline{x}(k) + \sum_{l=0,l\neq k}^{K-1} \overline{\overline{H}}(k,l)\,\overline{x}(l), ...k = 0,1,...,K-1 \tag{2.41}$$

Cette relation montre que la composante spectrale d'indice k du signal de sortie, $\overline{y}(k)$, est composée de deux parties :

- la première partie $\overline{\overline{H}}(k,k)\,\overline{x}(k)$ représente la composante linéaire invariante dans le temps (LIT). C'est la contribution de la composante spectrale d'indice k du signal d'entrée. Ainsi, les éléments $\overline{\overline{H}}(k,k)$ représentent les termes de la composante invariante dans le temps.
- La seconde partie $\sum_{l=0,l\neq k}^{K-1} \overline{\overline{H}}(k,l)\,\overline{x}(l)$ représente la composante de repliement de spectre (aliasing). C'est la contribution de toutes les autres composantes spectrales du signal d'entrée.

Ainsi, les éléments $\overline{H}(k,l)$, $k \neq l$ représententles termes du repliement de spectre.

Sachant que la distorsion d'un filtre correspond à l'erreur entre le filtrage obtenu et le filtrage désiré, celle-ci peut alors se décomposer en deux parties :

- la distorsion LIT dont les composantes sont données par :

$$e_i(k) = \left|\overline{\overline{H}}(k,k) - \overline{\overline{H_d}}(k,k)\right| \tag{2.42}$$

La distorsion LIT globale est donnée par :

$$e_i = \sum_{k=0}^{K-1} e_i(k) = \sum_{k=0}^{K-1} \left|\overline{\overline{H}}(k,k) - \overline{\overline{H_d}}(k,k)\right|^2 \tag{2.43}$$

- la distorsion de repliement du spectre dont les composantes sont données par :

$$e_r(k) = \sum_{l=0,\, l\neq k}^{K-1} \left|\overline{\overline{H}}(k,l) - \overline{\overline{H_d}}(k,l)\right|^2 \tag{2.44}$$

Le filtrage désiré étant toujours considéré comme un filtrage LIT, $\overline{\overline{H_d}}(k,l) = 0$ pour $l \neq k$, cette dernière relation (2.44) peut s'écrire :

$$e_r(k) = \sum_{l=0,\, l\neq k}^{K-1} \left|\overline{\overline{H}}(k,l)\right|^2 \tag{2.45}$$

L'erreur de repliement de spectre globale est donnée par :

$$e_r = \sum_{k=0}^{K-1} e_r(k) = \sum_{k=0}^{K-1} \left(\sum_{l=0,\, l\neq k}^{K-1} \left|\overline{\overline{H}}(k,l)\right|^2 \right) \tag{2.46}$$

La distorsion du filtre est la somme des deux distorsions, LIT et repliement de spectre. Ses composantes sont données par :

$$e(k) = e_i(k) + e_r(k) \quad k : 0, 1, ..., K-1 \tag{2.47}$$

soit,

$$e(k) = \sum_{l=0}^{K-1} \left|\overline{\overline{H}}(k,l) - \overline{\overline{H_d}}(k,l)\right|^2 \tag{2.48}$$

La distorsion totale est donnée par :

$$e = \sum_{k=0}^{K-1} e(k) = \sum_{k=0}^{K-1} \sum_{l=0}^{K-1} \left| \overline{\overline{H}}(k,l) - \overline{\overline{H_d}}(k,l) \right|^2 \tag{2.49}$$

soit,

$$e = \left\| \overline{\overline{H}} - \overline{\overline{H_d}} \right\|_f^2 \tag{2.50}$$

Pour concevoir le filtre bloc minimisant cette erreur de distorsion totale, nous allons décrire dans le paragraphe suivant quelques propriétés des matrices circulantes qui seront utilisées par la suite pour développer des algorithmes de très faible complexité de synthèse des filtres optimaux.

2.8 Matrices circulantes

Une matrice circulante est une matrice carrée dans laquelle on passe d'une ligne (colonne) à la suivante par permutation circulaire : décalage vers la droite (le bas) des coefficients. Dans ce paragraphe, nous rappelons quelques propriétés des matrices circulantes.

2.8.1 Définition

Une matrice circulante C d'ordre M est définie par $C(i,j) = c_{\langle j-i \rangle_M}$ $(i,j : 0,1,...,M-1)$:

$$C = \begin{pmatrix} c_0 & c_1 & c_2 & \cdots & c_{M-1} \\ c_{M-1} & c_0 & c_1 & & c_{M-2} \\ c_{M-2} & c_{M-1} & c_0 & & c_{M-3} \\ \vdots & & & \ddots & \vdots \\ c_1 & c_2 & c_3 & \cdots & c_0 \end{pmatrix} \tag{2.51}$$

où les c_i $(i : 0,1,...,M-1)$ peuvent être des réels ou des complexes. Un exemple d'une matrice circulante C d'ordre 4 est donné par :

$$C = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 1 & 2 & 3 \\ 3 & 4 & 1 & 2 \\ 2 & 3 & 4 & 1 \end{pmatrix}$$

Une matrice circulante est une matrice symétrique par rapport à sa diagonale secondaire, elle est dite matrice persymétrique :

$$C(i,j) = C(M-1-j, M-1-i), \quad i,j : 0,1,...,M-1 \tag{2.52}$$

2.8.2 Propriétés

L'ensemble des matrices circulantes de rang M forme un espace vectoriel de dimension M. La base de cet espace vectoriel est formée des matrices $P_i = P_M^i$ $(i : 0,1,...,M-1)$ où P_M est la matrice de permutation circulaire donnée par :

$$P_M = \begin{pmatrix} 0 & 1 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & 0 & \ddots & \ddots & 0 \\ 0 & 0 & \ddots & 0 & 1 \\ 1 & \cdots & 0 & 0 & 0 \end{pmatrix}$$

A cette matrice P_M est associée la transformation linéaire dite de "permutation circulaire" qui, à un vecteur $x = (x_0, x_1, ..., x_{M-1})$ fait correspondre le vecteur $y = (x_1, ..., x_{M-1}, x_0)$ obtenu par une permutation circulaire. P_M est aussi une matrice circulante possédant les propriétés suivantes :

- $P_M^M = I_M$
- P_M^i correspond à un décalage circulaire de i éléments.
- P_M est une matrice orthogonale : $P_M^{-1} = P_M^T$.

Les matrices circulantes possèdent les propriétés suivantes [Gray2006] :

- Étant données deux matrices circulantes A et B et un scalaire λ, les matrices $A+B$, AB et λA sont aussi des matrices circulantes
- Toute matrice circulante admet une matrice inverse qui est aussi circulante.

Une autre propriété très intéressante des matrices circulantes est celle de leur décomposition sur le corps des complexes $\mathbb{C}$. En effet, la diagonalisation des matrices circulantes fait intervenir la transformée de Fourier discrète selon l'équation :

$$C = F^{-1}DF$$

où D est une matrice diagonale telle que $diag\,(D) = FC\,(:,0)$, $C\,(:,0)$ étant la première colonne de la matrice C, F et F^{-1} désignent respectivement les matrices associées à la transformée de Fourier discrète directe et inverse.

C'est sur cette propriété, à la base de la réduction de la complexité de plusieurs algorithmes de traitement du signal, que nous nous appuierons pour développer les algorithmes rapides de synthèse des filtres optimaux.

2.9 Algorithmes rapides de synthèse des filtres optimaux

Tenant compte des structures diagonales des matrices rencontrées dans le filtrage OLS et OLA, nous nous basons sur les propriétés des matrices circulantes pour développer des algorithmes rapides de synthèse des filtres optimaux.

Étant donnée que $\overline{\overline{H_d}}$ est une matrice diagonale, H_d est alors une matrice circulante et un simple calcul de la transformée inverse de la diagonale de $\overline{\overline{H_d}}$ suffira pour obtenir les éléments de la première colonne de H_d :

$$H_d\,(:,0) = F_M^{-1} diag\left(\overline{\overline{H_d}}\right) \tag{2.53}$$

En raison de la structure particulière de la matrice H donnée dans les figures (2.2) et (2.4) et de la structure circulante de H_d, minimiser l'erreur e donnée par l'équation (2.40) revient à minimiser l'erreur e_1 donnée par :

$$e_1 = \|A - A_d\|_f^2 \tag{2.54}$$

où A est la matrice bloc qui compose la matrice H. La matrice A_d est la matrice bloc extraite de la matrice H_d de la même façon que A est extraite de H.

2.9.1 Algorithme rapide pour l'OLS

Dans le cas d'OLS, la minimisation de l'erreur e_1 donnée par l'équation (2.54) revient à minimiser l'erreur $e_{1,s}$ donnée par :

$$e_{1,s} = \|A_s - A_1\|_f^2$$

où A_1 est la matrice bloc extraite de la matrice H_d de la même façon que A_s est extraite de H_s.

En remplaçant la matrice A_s par son expression donnée par l'équation (2.25), cette erreur peut

s'écrire :

$$e_{1,s} = \left\| SF_M^{-1} G_s F_M - A_1 \right\|^2$$

soit,

$$e_{1,s} = \left\| SC_s - A_1 \right\|^2 \tag{2.55}$$

où C_s est la matrice donnée par :

$$C_s = F_M^{-1} G_s F_M \tag{2.56}$$

La norme de Frobenius d'une matrice A étant égale à la norme euclidienne du vecteur formé par la juxtaposition des lignes de cette matrice, l'erreur $e_{1,s}$ s'écrit :

$$e_{1,s} = \left\| ligne\left(SC_s - A_1\right) \right\|^2 \tag{2.57}$$

soit :

$$e_{1,s} = \left\| ligne\left(SC_s\right) - ligne\left(A_1\right) \right\|^2 \tag{2.58}$$

où $ligne\left(A\right)$ désigne le vecteur obtenu par la juxtaposition des lignes de la matrice A. G_s étant diagonale, la matrice C_s est alors une matrice circulante. Par conséquent, les lignes de la matrice C_s sont liées par la relation :

$$C_s\left(i,:\right)^T = P_M C_s\left(i-1,:\right)^T ; \;\; \left(i : 0, 1, ..., M-1\right) \tag{2.59}$$

En exprimant le vecteur $ligne\left(SC_s\right)$ en fonction de la première ligne de la matrice C_s, nous obtenons :

$$e_{1,s} = \left\| \begin{bmatrix} \left(P_M\right)^d \\ \left(P_M\right)^{d+1} \\ \vdots \\ \left(P_M\right)^{M-d-1} \end{bmatrix} C_s\left(0,:\right)^T - \begin{bmatrix} A_1\left(0,:\right)^T \\ A_1\left(1,:\right)^T \\ \vdots \\ A_1\left(L-1,:\right)^T \end{bmatrix} \right\|^2 \tag{2.60}$$

Ainsi, le vecteur colonne $C_s\left(0,:\right)^T$, formé par les éléments de la première ligne de C_s, qui minimise

cette erreur $e_{1,s}$ peut être obtenu par la méthode de pseudo-inverse :

$$C_s(0,:)^T = \left(\begin{bmatrix} (P_M)^d \\ (P_M)^{d+1} \\ \vdots \\ (P_M)^{M-d-1} \end{bmatrix}^T \begin{bmatrix} (P_M)^d \\ (P_M)^{d+1} \\ \vdots \\ (P_M)^{M-d-1} \end{bmatrix} \right)^{-1} \begin{bmatrix} (P_M)^d \\ (P_M)^{d+1} \\ \vdots \\ (P_M)^{M-d-1} \end{bmatrix}^T \begin{bmatrix} A_1(0,:)^T \\ A_1(1,:)^T \\ \vdots \\ A_1(L-1,:)^T \end{bmatrix} \tag{2.61}$$

Les matrices permutation $(P_M)^i$ $(i : d \to M-d-1)$ possèdent les propriétés suivantes :

$$\left((P_M)^i\right)^T = \left((P_M)^i\right)^{-1} = \left(P_M^{-1}\right)^i$$

$$\left((P_M)^i\right)^T (P_M)^i = I_M$$

L'utilisation de ces deux dernières propriétés permet d'écrire :

$$C_s(0,:)^T = \frac{1}{L} \sum_{i=0}^{L-1} \left(\left(P_M^{-1}\right)^{i+d} A_1(i,:)^T \right) \tag{2.62}$$

soit,

$$C_s(0,j) = \frac{1}{L} \sum_{i=0}^{L-1} A_1(i, \langle i+j+d \rangle_M) \tag{2.63}$$

Notons que la matrice $\left(P_M^{-1}\right)$ est la matrice de permutation correspondant à un décalage cyclique à gauche. L'application de la matrice $\left(P_M^{-1}\right)^i$ sur un vecteur X consiste à effectuer, sur les éléments de ce vecteur, une permutation circulaire de i éléments dans le sens de bas en haut, ce qui revient, pour le vecteur X^T à une permutation circulaire à gauche de i éléments. Ainsi aucune opération d'addition ou de multiplication n'est nécessaire :

$$\left(P_M^{-1}\right)^i [x_0, ..., x_i, x_{i+1}, ..., x_{M-1}]^T = [x_i, ..., x_{M-1}, x_0, ..., x_{i-1}]^T$$

C_s étant une matrice circulante, les éléments de la diagonale de G_s sont obtenus par la relation :

$$diag(G_s) = M F_M^{-1} C_s(0,:)^T \tag{2.64}$$

Ces éléments peuvent aussi être obtenus par la relation :

$$diag\left(G_s\right) = F_M C_s\left(:,0\right) \tag{2.65}$$

où $C_s\left(:,0\right)$ est la première colonne de C_s dont les éléments sont liés à ceux de la première ligne par :

$$C_s\left(i,0\right) = C_s\left(0,M-i\right), \;\; \left(i:0,1,...,M-1\right)$$

Ainsi les coefficients de la matrice diagonale G_s qui minimise l'erreur $e_{1,s}$ sont ceux donnés par l'une des relations précédentes.

2.9.2 Algorithme rapide pour l'OLA

Dans ce cas d'OLA, la minimisation de l'erreur e_1 donnée par l'équation (2.54) revient à minimiser l'erreur $e_{1,a}$ donnée par :

$$e_{1,a} = \left\|A_a - A_2\right\|_f^2$$

où A_2 est la matrice bloc extraite de la matrice H_d de la même façon que A_a est extraite de H_a.

En remplaçant la matrice A_a par son expression donnée par l'équation (2.32), cette erreur peut s'écrire :

$$e_{1,a} = \left\|F_M^{-1} G_a F_M S^T - A_2\right\|^2$$

soit,

$$e_{1,a} = \left\|C_a S^T - A_2\right\|^2 \tag{2.66}$$

où C_a est la matrice donnée par :

$$C_a = F_M^{-1} G_a F_M \tag{2.67}$$

La norme de Frobenius d'une matrice A est égale à la norme euclidienne du vecteur formé par la juxtaposition des colonnes de cette matrice, l'erreur $e_{1,a}$ s'écrit :

$$e_{1,a} = \left\|col\left(C_a S^T - A_2\right)\right\|^2 \tag{2.68}$$

soit,

$$e_{1,a} = \left\|col\left(C_a S^T\right) - col\left(A_2\right)\right\|^2 \tag{2.69}$$

où $col\,(A)$ désigne le vecteur obtenu par la juxtaposition des colonnes de la matrice A.

G_a étant diagonale, la matrice C_a est alors une matrice circulante. Par conséquent, les colonnes de la matrice C_a sont liées par la relation :

$$C_a\,(:,i) = P_M C_a\,(:,i-1)\,; \quad (i:0,1,...,M-1) \tag{2.70}$$

En exprimant le vecteur $col\left(C_a S^T\right)$ en fonction de la première colonne de la matrice C_a, nous obtenons :

$$e_{1,a} = \left\| \begin{bmatrix} (P_M)^d \\ (P_M)^{d+1} \\ \vdots \\ (P_M)^{M-d-1} \end{bmatrix} C_a\,(:,0) - \begin{bmatrix} A_2\,(:,0) \\ A_2\,(:,1) \\ \vdots \\ A_2\,(:,L-1) \end{bmatrix} \right\|^2 \tag{2.71}$$

Ainsi, le vecteur colonne $C_a\,(:,0)$ formé par les éléments de la première colonne de C_a, qui minimise cette erreur $e_{1,a}$, peut être obtenu par la méthode de pseudo-inverse :

$$C_a\,(:,0) = \left(\begin{bmatrix} (P_M)^d \\ (P_M)^{d+1} \\ \vdots \\ (P_M)^{M-d-1} \end{bmatrix}^T \begin{bmatrix} (P_M)^d \\ (P_M)^{d+1} \\ \vdots \\ (P_M)^{M-d-1} \end{bmatrix} \right)^{-1} \begin{bmatrix} (P_M)^d \\ (P_M)^{d+1} \\ \vdots \\ (P_M)^{M-d-1} \end{bmatrix}^T \begin{bmatrix} A_2\,(:,0) \\ A_2\,(:,1) \\ \vdots \\ A_2\,(:,L-1) \end{bmatrix} \tag{2.72}$$

Ainsi, de la même manière que celle décrite dans le cas d'OLS, l'utilisation les propriétés des matrices permutation $(P_M)^i$ permet d'écrire :

$$C_a\,(:,0) = \frac{1}{L} \sum_{i=0}^{L-1} \left(\left(P_M^{-1}\right)^{i+d} A_2\,(:,i) \right) \tag{2.73}$$

soit,

$$C_a\,(i,0) = \frac{1}{L} \sum_{j=0}^{L-1} A_2\left(\langle i+j+d \rangle_M\,, j\right) \tag{2.74}$$

Les éléments de la diagonale de G_a peuvent être alors obtenus par :

$$diag\,(G_a) = F_M C_a\,(:,0) \tag{2.75}$$

Ainsi les coefficients de la matrice diagonale G_a qui minimise l'erreur $e_{1,a}$ sont ceux donnés par cette

relation.

2.9.3 Coût de synthèse

La première étape de synthèse du filtre bloc optimal est le calcul de la matrice H_d de laquelle sont extraites les matrices A_1 ou A_2 . En exploitant les propriétés des matrices circulantes, cela nécessite un calcul TFD $K-$points, soit un nombre de multiplications réelles de l'ordre de $Klog_2K$. La seconde étape qui est celle du calcul des éléments des matrices circulantes C_s ou C_a, ne nécessite aucune opération de multiplication. La dernière étape, qui est celle du calcul de la diagonale de G_s ou de G_a, nécessite un calcul TFD $M-$points, soit un nombre de multiplications réelles de l'ordre de $Mlog_2M$.

Au total, cette méthode de synthèse de filtre bloc a une complexité de l'ordre de $(Klog_2K + Mlog_2M)$ multiplications réelles.

2.9.4 Algorithme rapide de conception d'un filtre bloc LIT optimal

Les filtres blocs, obtenus par les algorithmes rapides développés et répondant à un problème d'optimisation sans aucune contrainte sur les inconnues, sont en général des filtres LPVT. Ainsi, pour des applications pour lesquelles le repliement de spectre n'est pas autorisé, ces filtres ne peuvent pas être utilisés. Notre objectif dans ce paragraphe est de dériver les algorithmes rapides permettant d'obtenir un filtrage LIT optimal.

Sachant que dans le cas d'un filtrage LIT les erreurs de repliement de spectre sont nulles, $\overline{\overline{H_s}}$ et $\overline{\overline{H_a}}$ doivent être ainsi des matrices diagonales, et parsuite, H_s et H_a sont des matrices circulantes. En se référant aux figures (2.2)et (2.4), H_s et H_a sont circulantes si :

$$A_s(i,j) = 0 \quad pour \quad |j-i-d| > d \tag{2.76}$$

$$A_a(i,j) = 0 \quad pour \quad |i-j-d| > d \tag{2.77}$$

En tenant compte des relations $A_s = SC_s$ et $A_a = C_aS^T$, les 2 équations (2.76) et (2.77) sont valables si :

$$C_s(0,j) = 0 \quad pour \quad d < j < M-d \tag{2.78}$$

$$C_a(i,0) = 0 \quad pour \quad d < i < M-d \tag{2.79}$$

Ces deux dernières relations sont équivalentes à la condition $P \leq M-L$ posée dans un problème de filtrage OLS ou OLA LIT classique. En effet, la relation $P \leq M-L$, qui signifie que $h(n)=0$ pour $n > M-L = 2d$, est requise dans le cas d'un filtrage causal. Le filtrage causal introduisant un retard est nécessaire uniquement dans le cas d'une réalisation directe du filtrage RIF. Dans le cas de notre filtrage OLS par exemple, la relation $y_s(n) = \sum_{i=0}^{M-1} A_s(\langle n \rangle_L, i)\, x(n-d+i)$ montre que le filtrage obtenu est non causal puisque les échantillons $y_s(n)$ du signal de sortie peuvent dépendre des échantillons $x(k)$ tel que $k > n$. Ainsi, pour un filtrage non causal, la contrainte LIT s'écrit $h(n)=0$ pour $|n| > d$. D'après la propriété de périodicité de la transformée de Fourier discrète de taille M, nous pouvons en déduire que $h(n)=0$ pour $d < n < M-d$.

D'après les équations (2.60) et (2.66), nous pouvons distinguer que les éléments de $C_s(0,:)$ ou $C_a(:,0)$ à optimiser sont des variables indépendantes. Il en résulte que les éléments de $C_s(0,:)$ ou $C_a(:,0)$ laissés libres par les conditions d'invariance dans le temps (2.78) ou (2.79) possèdent les mêmes valeurs optimales obtenues dans le cas de l'optimisation sans contrainte. Ainsi, nous obtenons :

$$\begin{cases} C_s(0,j) &= 0 & pour\ d+1 \leq j \leq M-d-1 \\ C_s(0,j) &= \frac{1}{L}\sum_{i=0}^{L-1} A_1(i, \langle i+j+d \rangle_M) & ailleurs \end{cases}$$

$$\begin{cases} C_a(i,0) &= 0 & pour\ d+1 \leq i \leq M-d-1 \\ C_a(i,0) &= \frac{1}{L}\sum_{j=0}^{L-1} A_2(\langle i+j+d \rangle_M, j) & ailleurs \end{cases}$$

Le filtrage ainsi obtenu par notre algorithme de conception optimal des filtres LIT est le même que celui que nous obtenons par la méthode classique des moindres carrés (fonction 'FIRLS' dans MATLAB). L'avantage de notre algorithme de synthèse réside donc dans son faible coût de calcul.

2.10 Comparaison des conceptions optimales OLS et OLA

Pour des filtres invariants dans le temps, les deux implémentations OLS et OLA ne présentent pas de différence. L'invariance temporelle est obtenue dès que l'ordre P du filtre et les tailles des blocs M et L sont tels que $P \leq M-L$. Le fait d'implémenter un filtre LIT en OLS ou en OLA ne présente aucun effet ni sur les erreurs de distorsions globales ni sur leur distribution par rapport aux différentes fréquences.

Cependant, les méthodes de synthèse des filtres blocs optimaux développées dans ce chapitre aboutissent en général à des filtres LPVT. Ainsi, la réponse en fréquence possède deux composantes : la composante invariante dans le temps et la composante de repliement de spectre. En outre, comme nous avons déjà vu dans l'algorithme rapide de conception optimale, le développement des calculs est différent selon le type de choix d'implémentation.

Dans la suite, nous démontrerons que, dans le cas d'OLS et d'OLA, la conception optimale aboutit aux mêmes valeurs des éléments de la matrice G et aux mêmes erreurs de distorsions globales. Nous démontrons aussi que, les composantes de l'erreur invariante dans le temps sont identiques alors que les composantes de repliement de spectre sont différentes.

2.10.1 Comparaison des matrices G_a et G_s

Dans cette partie, nous démontrons que les matrices G qui représentent les filtres blocs optimaux en OLS et OLA sont égales, $G_a = G_s$. En effet, les éléments des matrices A_1 et A_2, extraites respectivement de la matrice H_d selon les structures montrées en figure (2.2)et (2.4) sont données par :

$$A_1(i,j) = H_d(i, \langle j-d \rangle_K), \quad (i: 0 \to L-1,\, j: 0 \to M-1)$$

$$A_2(i,j) = H_d(\langle i-d \rangle_K, j) \quad (i: 0 \to M-1,\, j: 0 \to L-1)$$

La matrice H_d étant une matrice circulante, les éléments de A_1 et A_2 sont alors liés par la relation :

$$A_1(i,j) = A_2(M-1-j, L-1-i) \quad (i: 0 \to L-1,\, j: 0 \to M-1)$$

En tenant compte de cette dernière relation, nous déduisons à partir des équations (2.63) et (2.74) que :

$$C_s(0,i) = C_a(M-i, 0)$$

Les matrices C_s et C_a étant circulantes, nous pouvons écrire :

$$C_a(\langle M-i+m \rangle_M, \langle 0+m \rangle_M) = C_a(M-i, 0)$$

Soit pour $m = i - M$,

$$C_a(0,i) = C_a(M-i,0)$$

Ainsi,

$$C_s(0,i) = C_a(0,i)$$

Les deux matrices C_s et C_a, et par conséquent, les deux matrices diagonales G_s et G_a sont donc égales.

Notons deux cas particuliers utiles pour les applications pratiques :

- cas où la diagonale de la matrice diagonale $\overline{\overline{H_d}}$ est un vecteur réel : la matrice H_d est une matrice Hermitienne ($H_d = H_d^\star$). Les matrices C_s et C_a obtenues sont alors des matrices circulantes Hermitiennes et les matrices G_s et G_a sont diagonales à coefficients réels.
- cas où la diagonale de la matrice diagonale $\overline{\overline{H_d}}$ est un vecteur et symétrique : la matrice H_d est une matrice symétrique ($H_d = H_d^T$). Les matrices C_s et C_a obtenues sont alors des matrices circulantes symétriques et les matrices G_s et G_a sont diagonales à coefficients réels symétriques.

2.10.2 Comparaison des erreurs de distorsion

Les matrices C_s et C_a étant égales, ainsi les éléments des matrices $A_s = SC_s$ et $A_a = C_a S^T$ sont liés par :

$$A_s(i,j) = A_a(M-1-j, L-1-i), \quad (i: 0 \to L-1,\ j: 0 \to M-1)$$

Par conséquent, les matrices H_s et H_a sont telles que :

$$H_a(i,j) = H_s(K-1-j, K-1-i), \quad (i,j: 0 \to K-1) \tag{2.80}$$

En développant l'expression de l'erreur de distorsion globale, nous obtenons pour les deux cas, OLS et OLA, les expressions suivantes :

$$e_s = \|H_d - H_s\|_f^2 = \sum_{i=0}^{K-1}\sum_{j=0}^{K-1}\left(|H_d(i,j) - H_s(i,j)|^2\right)$$

$$e_a = \|H_d - H_a\|_f^2 = \sum_{i=0}^{K-1}\sum_{j=0}^{K-1}\left(|H_d(i,j) - H_a(i,j)|^2\right)$$

En posant $i' = K-1-j$ et $j' = K-1-i$, et en utilisant la relation (2.80) , l'erreur e_s s'écrit :

$$e_s = \sum_{j'=0}^{K-1} \sum_{i'=0}^{K-1} \left(\left| H_d\left(i', j'\right) - H_a\left(i', j'\right) \right|^2 \right)$$

soit,

$$e_s = e_a$$

Ainsi, l'erreur globale de distorsion est la même pour les deux approches OLS et OLA.

Pour comparer la distribution en fréquences de l'erreur globale de distorsion, rappelons que celle-ci est formée de deux parties : la distorsion LIT et la distorsion de repliement de spectre. L'erreur de repliement de spectre est définie comme étant l'erreur entre les éléments non diagonaux de $\overline{\overline{H_s}}$ et $\overline{\overline{H_d}}$ ou de $\overline{\overline{H_a}}$ et $\overline{\overline{H_d}}$. L'erreur de la réponse fréquentielle invariante dans le temps est définie comme étant l'erreur entre les éléments diagonaux.

Les composantes de la distorsion LIT

Les composantes fréquentielles de l'erreur de distorsion LIT dans les deux cas OLS et OLA sont respectivement données par :

$$e_{i,s}(k) = \left| \overline{\overline{H_d}}\left(k,k\right) - \overline{\overline{H_s}}\left(k,k\right) \right|^2$$

$$e_{i,a}(k) = \left| \overline{\overline{H_d}}\left(k,k\right) - \overline{\overline{H_a}}\left(k,k\right) \right|^2$$

D'après les équations (2.28)et (2.35), les éléments des matrices $\overline{\overline{H_s}}$ et $\overline{\overline{H_a}}$ s'écrivent respectivement :

$$\overline{\overline{H_s}}\left(i,j\right) = \sum_{m=0}^{K-1} \sum_{n=0}^{K-1} F_K\left(i,m\right) H_s\left(m,n\right) F_K^{-1}\left(n,j\right)$$

$$\overline{\overline{H_a}}\left(i,j\right) = \sum_{m=0}^{K-1} \sum_{n=0}^{K-1} F_K\left(i,m\right) H_a\left(m,n\right) F_K^{-1}\left(n,j\right)$$

Sachant que $F_K\left(i,m\right) F_K^{-1}\left(n,j\right) = \frac{\alpha_K^{-im}\alpha_K^{nj}}{K} = \frac{\alpha_K^{-im+jn}}{K}$, $\alpha_K = e^{j\frac{2\pi}{K}}$ étant la racine $K^{ème}$ de l'unité, nous obtenons :

$$\overline{\overline{H_s}}\left(i,j\right) = \frac{1}{K} \sum_{m=0}^{K-1} \sum_{n=0}^{K-1} \alpha_K^{-im+jn} H_s\left(m,n\right)$$

$$\overline{\overline{H_a}}(i,j) = \frac{1}{K}\sum_{m=0}^{K-1}\sum_{n=0}^{K-1}\alpha_K^{-im+jn}H_a(m,n)$$

En utilisant la relation entre les éléments de H_s et H_a définie dans l'équation (2.80), les éléments $\overline{\overline{H_s}}(i,j)$ s'écrivent :

$$\overline{\overline{H_s}}(i,j) = \frac{1}{K}\sum_{m=0}^{K-1}\sum_{n=0}^{K-1}\alpha_K^{-im+jn}H_a(K-1-n,K-1-m)$$

En posant $m' = K-1-n$ et $n' = K-1-m$, nous obtenons :

$$\overline{\overline{H_s}}(i,j) = \frac{\alpha_K^{i-j}}{K}\sum_{m'=0}^{K-1}\sum_{n'=0}^{K-1}\alpha_K^{in'-jm'}H_a(m',n')$$

Soit,

$$\overline{\overline{H_s}}(i,j) = \alpha_K^{i-j}\overline{\overline{H_a}}(j,i) \tag{2.81}$$

Pour $i = j$, nous déduisons que les termes diagonaux des matrices $\overline{\overline{H_s}}$ et $\overline{\overline{H_a}}$ sont égaux :

$$\overline{\overline{H_s}}(i,i) = \overline{\overline{H_a}}(i,i), \quad i = 0,1,...K-1$$

Ainsi, les composantes de l'erreur de distorsion LIT sont les mêmes pour les deux approches OLS et OLA :

$$e_{i,s}(k) = e_{i,a}(k), \quad k = 0,1,...,K-1$$

Par suite, l'erreur globale de distorsion LIT est la même pour les deux cas OLS et OLA :

$$e_{i,s} = e_{i,a}$$

Les composantes de repliement de spectre

Étant donné que les deux approches OLS et OLA fournissent la même valeur de l'erreur de distorsion totale et aussi la même valeur de l'erreur de distorsion LIT, l'erreur de repliement de spectre a aussi la même valeur pour les deux approches.

Dans le cas des filtres LIT, l'erreur de repliement de spectre étant nulle, ces composantes en fréquences sont toutes nulles. Par conséquence, les composantes de repliement de spectre sont identiques pour les deux approches OLS et OLA.

Cependant, dans le cas des filtres blocs LPVT, bien que cet erreur de repliement de spectre ait la même valeur, ses composantes en fréquences diffèrent pour les deux approches OLS et OLA.

Sachant qu'un élément $\overline{\overline{H_s}}(i,j)$ ou $\overline{\overline{H_a}}(i,j)$, $i \neq j$, représente l'effet de repliement de spectre de la composante spectrale d'entrée $\overline{x}(j)$ sur la composante spectrale de sortie $\overline{y}(i)$, nous obtenons les résultats suivants :

Propriété 2.4

Les composantes de repliement de spectre dans le cas d'OLS et celles dans le cas d'OLA sont liées par :

$$\left|\overline{\overline{H_s}}(i,j)\right| = \left|\overline{\overline{H_a}}(j,i)\right|$$

Cette propriété est déduite de l'équation (2.81).

Propriété 2.5

Le repliement de spectre d'une fréquence de sortie i peut être causé uniquement par les fréquences d'entrée j telles que :

$$j - i = q\frac{K}{L}, \quad q \in \mathbb{Z}^{\star}$$

soit,

$$\forall\,(i,j)\,, \left(j - i \neq q\frac{K}{L},\ q \in \mathbb{Z}^{\star}\right) \Rightarrow \begin{cases} \overline{\overline{H_s}}(i,j) &= 0 \\ \overline{\overline{H_a}}(i,j) &= 0 \end{cases}$$

La démonstration de cette propriété est donnée en *Annexe A*.

Propriété 2.6

En OLS, les fréquences d'entrée j telles que $j = t\frac{K}{pgcd(K,M)}$, $t \in \mathbb{Z}$, n'ont pas d'effet de repliement de spectre sur aucune fréquence de sortie i, soit :

$$\forall\, i \neq j,\ \left(j = t\frac{K}{pgcd\,(K,M)},\ t \in \mathbb{Z}\right) \Rightarrow \overline{\overline{H_s}}(i,j) = 0$$

En OLA, les fréquences de sortie i telles que $i = t\frac{K}{pgcd(K,M)}$, $t \in \mathbb{Z}$, ne sont pas soumises à des effetes de repliement de spectre par aucune fréquence d'entrée j, soit :

$$\forall\, i \neq j,\ \left(i = t\frac{K}{pgcd\,(K,M)},\ t \in \mathbb{Z}\right) \Rightarrow \overline{\overline{H_a}}\,(i,j) = 0$$

$pgcd\,(K,M)$ est le plus grand commun diviseur des entiers K et M.

La démonstration de cette propriété est donnée en *Annexe B*.

2.11 Comparaison entre l'approche optimale et les approches classiques

Pour illustrer notre approche de conception optimale, nous présentons et interprétons les résultats des simulations réalisées pour des tailles de blocs $M = 32$ et $L = 24$ et une résolution fréquentielle $\theta = \frac{F_e}{K} = \frac{F_e}{96}$. Ces valeurs de M, L et K permettent une visualisation claire des différents vecteurs et matrices obtenus. Des résultats équivalents pourront être obtenus avec des valeurs relativement supérieures correspondant à des applications pratiques.

Le filtre désiré choisi est un filtre passe bas, de fréquence de coupure $F_c = 10F_e/96$ où F_e est la fréquence d'échantillonnage. Pour une résolution fréquentielle $\theta = F_e/96$, la réponse fréquentielle du filtre désiré donnée par la figure 2.5 est représentée par un vecteur de taille 96 dont les coefficients sont nuls entre les indices 11 et 85 et égaux à 1 ailleurs. Les résultats des simulations sont représentés sur les figures 2.6 à 2.16.

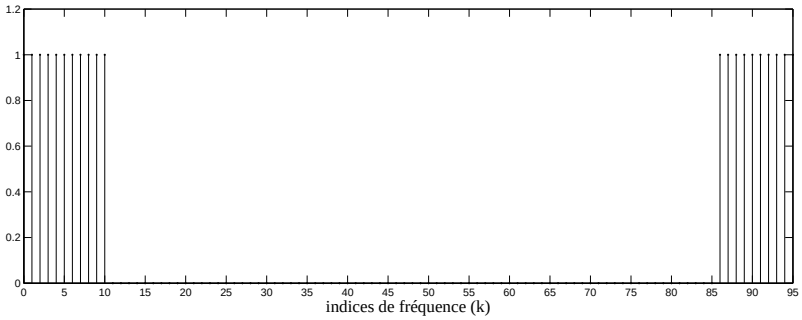

FIGURE 2.5 – Réponse en fréquence du filtre désiré

Nous présentons tout d'abord les diagonales des matrices G obtenues dans le cas de synthèse d'un filtre LIT selon notre approche de conception optimale et selon les approches classiques. Les

résultats de ces simulations sont représentés sur les figures 2.6, 2.7 et 2.8. Nous rappelons que le filtrage par blocs est un filtrage LIT si l'ordre P du filtre est inférieur ou égal à $M - L = 2d$, soit $P \leq 8$ dans le cas de nos simulations. Les approches classiques utilisées sont celles de fenêtrage utilisant la fenêtre de Hamming et la fenêtre de Kaiser, celles d'optimisation par la méthode des moindres carrés et la méthode de Parks-McClellan.

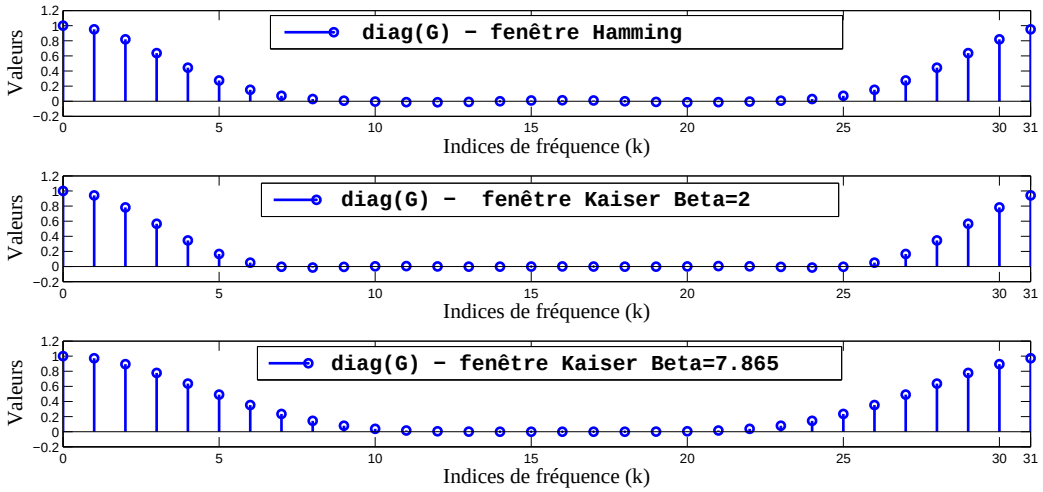

FIGURE 2.6 – Synthèse des filtres LIT selon la méthode des fenêtres

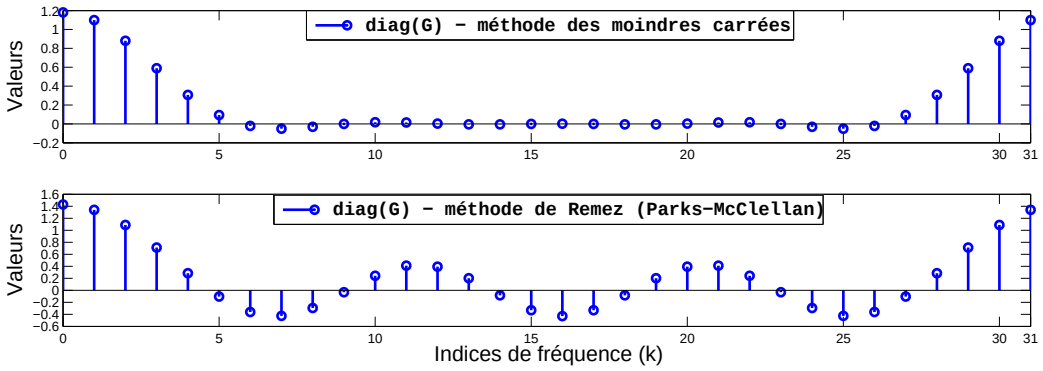

FIGURE 2.7 – Diagonale de G : Synthèse des filtres LIT selon les méthodes d'optimisation

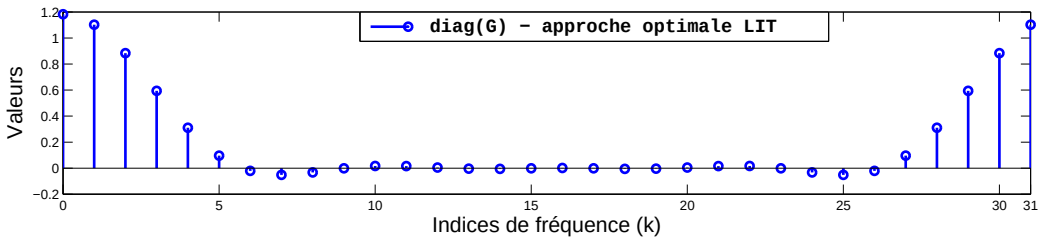

FIGURE 2.8 – Diagonale de G : Synthèse du filtre LIT selon notre approche optimale

Pour le cas du filtrage par blocs LPVT, les figures 2.9 et 2.10 représentent respectivement les diagonales de G obtenues selon l'approche classique d'échantillonnage en fréquence et selon notre approche optimale.

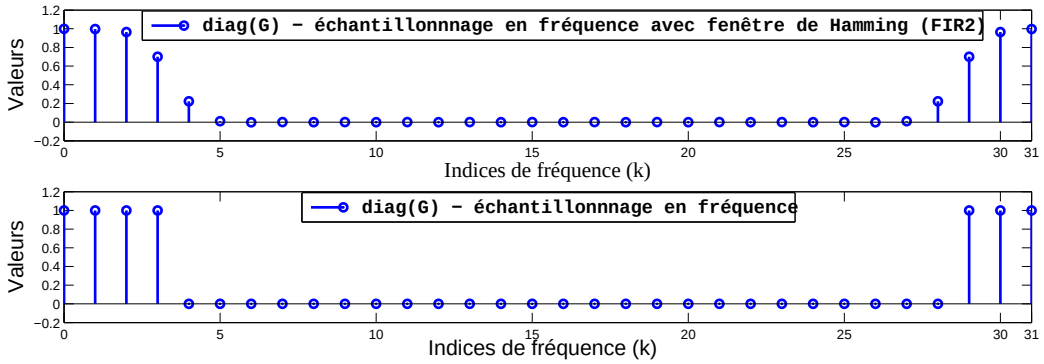

FIGURE 2.9 – Diagonale de G : Synthèse des filtres selon la méthode d'échantillonnage en fréquence

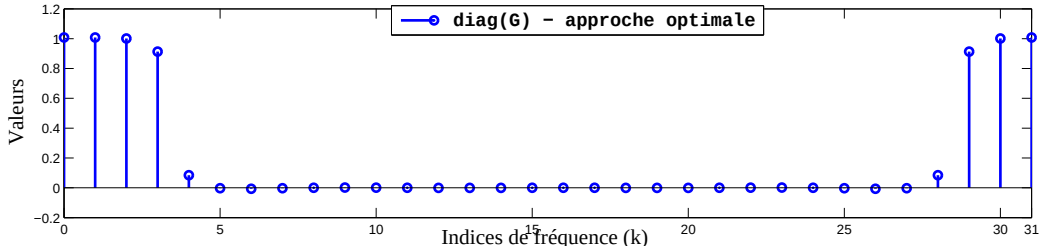

FIGURE 2.10 – Diagonale de G : Synthèse du filtre LPVT selon notre approche optimale

Nous comparons ensuite les différentes méthodes en terme d'erreur de distorsion. Pour chaque méthode, nous donnons :

- L'erreur de distorsion LIT définie selon l'équation (2.43).
- L'erreur de repliement de spectre (Aliasing) définie selon l'équation (2.46)
- L'erreur de distorsion globale définie selon l'équation (2.50).

Le tableau (2.1) montre les résultats de comparaison des différentes erreurs selon les différentes méthodes de synthèse des filtres RIF. Dans le cas où aucune contrainte n'est imposée sur la synthèse du filtre, notre approche optimale permet de concevoir le filtre bloc ayant la moindre erreur de distorsion globale et le filtrage obtenu est LPVT. Dans le cas où la contrainte d'invariance temporelle est imposée, le filtre bloc obtenu par notre approche optimale est identique à celui obtenu par la méthode classique de synthèse par moindres carrés. Cependant, l'avantage de notre méthode réside dans son faible coût de mise en oeuvre. Bien que cette conception LIT présente une erreur de repliement de spectre nulle, l'erreur de distorsion globale reste importante comparée à celle obtenue

par l'approche optimale sans contrainte. Cela montre que le fait de tolérer un léger repliement de spectre permet d'obtenir un gain considérable sur la qualité globale du filtre obtenu.

TABLE 2.1 – Comparaison des erreurs de distorsion selon les différentes méthodes de synthèse des filtres

Méthode		LIT	Aliasing	distorsion globale
Filtrage LIT	Fenêtrage Hamming	3.783	0	3.783
	Fenêtrage Kaiser ($\beta = 2$)	2.434	0	2.434
	Fenêtrage Kaiser ($\beta = 7.865$)	5.559	0	5.559
	Parks-McClellan (Remez)	8.711	0	8.711
	Moindres carrés	2.024	0	2.024
	Notre approche optimale	2.024	0	2.024
Filtrage LPVT	Échantillonnage fréquentiel	0.325	0.558	0.883
	Échantillonnage fréquentiel+ fenêtre Hamming	1.7107	0.0815	1.7922
	Notre approche optimale	0.4647	0.3291	0.7938

Les figures (2.11) et (2.12) représentent en 3D les matrices $\overline{\overline{H}}$ obtenues par notre approche de conception optimale développée dans ce chapitre. Sur ces figures, la hauteur représente la valeur de la racine carrée du module des éléments de la matrice, les éléments de la diagonale principale représentent les termes LIT alors que les éléments hors de cette diagonale représentent le repliement de spectre. Pour la figure (2.11), la matrice $\overline{\overline{H}}$ est diagonale, l'erreur de repliement de spectre est alors nulle ce qui correspond à notre hypothèse de filtrage LIT. Cependant pour la figure (2.12), la matrice $\overline{\overline{H}}$ obtenue n'est pas diagonale, ainsi l'erreur de repliement de spectre n'est pas nulle.

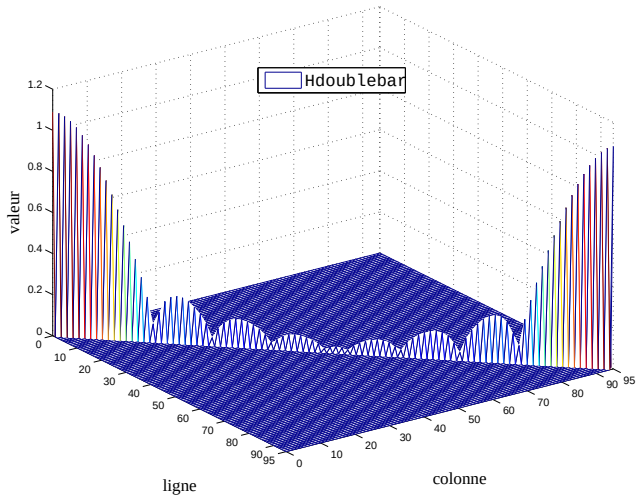

FIGURE 2.11 – Représentation 3D de la matrice $\overline{\overline{H}}$ du filtre optimal LIT

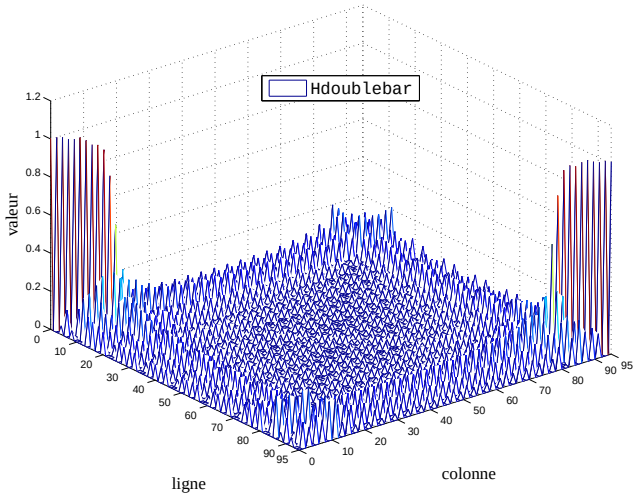

FIGURE 2.12 – Représentation 3D de la matrice $\overline{\overline{H}}$ du filtre optimal LPVT

Les figures (2.13) et (2.14) représentent en 2D les matrices $\overline{\overline{H}}$ représentant les réponses fréquentielles des filtres blocs OLS selon l'approche optimale LPVT et l'approche optimale LIT. Pour le cas général du filtrage LPVT (figure (2.13)), nous constatons que $\overline{\overline{H}}(i,j) = 0$ quels que soit i et j tel que $j - i \neq q\frac{K}{L} = 4q$, $q \in \mathbb{Z}^\star$, et que $\overline{\overline{H}}(i,j) = 0$ quel que soit $j = t\frac{K}{pgcd(K,M)} = 3t$, $t \in \mathbb{Z}$, $i \neq j$. Ces résultats vérifient les propriétés 2.5 et 2.6. Pour le cas de filtrage LIT (figure (2.14)), nous ne représentons que la diagonale de la matrice $\overline{\overline{H}}$ puisque tous les autres coefficients sont nuls.

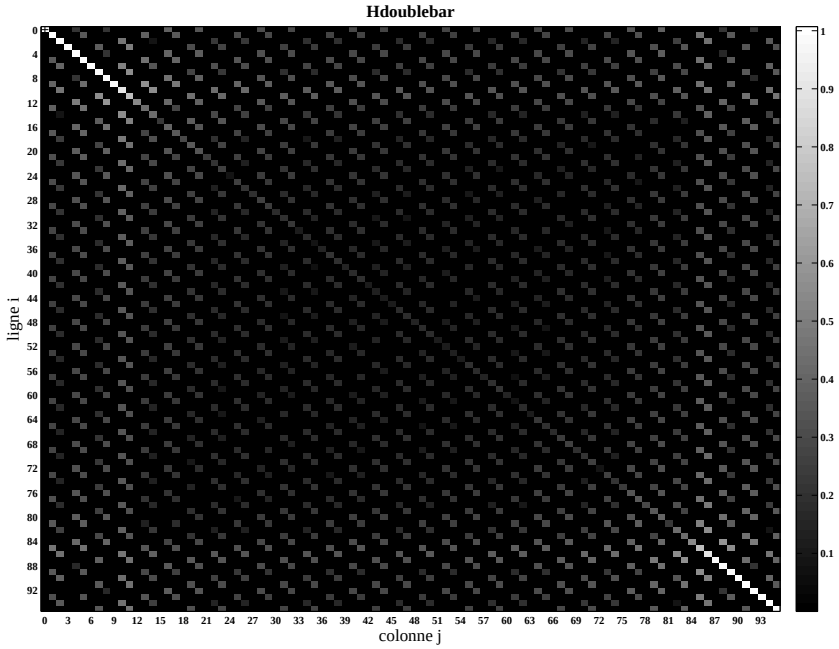

FIGURE 2.13 – Représentation 2D de la matrice $\overline{\overline{H}}$ du filtre optimal LPVT

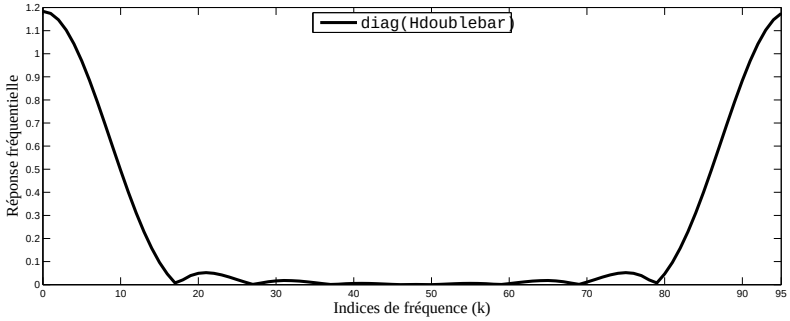

FIGURE 2.14 – $diag\left(\overline{\overline{H}}\right)$: Réponse fréquentielle du filtre optimal LIT

Les figures (2.15) et (2.16) montrent, en échelle logarithmique, les erreurs de distorsion pour les différentes fréquences. Nous pouvons constater que les pics des erreurs sont localisés dans les bandes de fréquence de transistion.

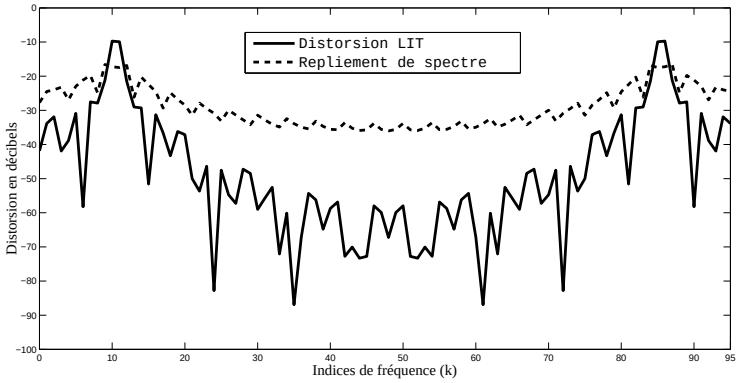

FIGURE 2.15 – Distorsion du filtre optimal LPVT

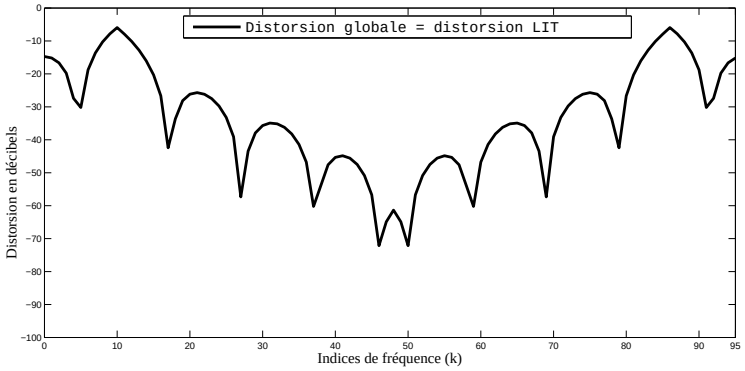

FIGURE 2.16 – Distorsion du filtre optimal LIT

2.12 Conclusion

L'overlap-save et l'overlap-add sont deux méthodes de filtrage par blocs permettant de réduire le coût de filtrage numérique par l'utilisation de la transformée de Fourier rapide. Dans ce chapitre, nous avons développé les modèles mathématiques de ces deux techniques de filtrage. Nous avons ensuite proposé une nouvelle approche de conception optimale des filtres blocs basée sur la minimisation de l'erreur quadratique. Utilisant les propriétés des matrices circulantes, nous avons enfin développé un algorithme de synthèse de très faible complexité.

Dans le chapitre suivant, nous traitons le même problème mais avec une généralisation du critère d'optimisation d'une part, de la transformée utilisée d'autre part, et aussi du traitement par blocs.

Chapitre 3

Conception généralisée optimale des filtres blocs

3.1 Introduction

Dans le chapitre précédent, un algorithme de synthèse optimale des filtres blocs OLS et OLA a été développé en nous basant sur la structure matricielle du filtrage OLS et OLA classique. Dans la structure matricielle développée, le filtre bloc était représenté par une matrice diagonale. Le filtre bloc optimal était obtenu par une minimisation de l'erreur quadratique moyenne entre le filtrage désiré et le filtrage par blocs.

Dans ce chapitre, nous proposons une nouvelle conception, dite généralisée, des méthodes de filtrage par blocs OLS et OLA. Dans ce nouveau modèle, le filtre bloc est représenté par une matrice quelconque et non nécessairement par une matrice diagonale. La structure classique d'OLS et d'OLA n'est alors qu'un cas particulier de ce modèle général lorsque la matrice est diagonale. Pour la synthèse du filtre généralisé optimal, nous utilisons le même critère d'optimisation par minimisation de l'erreur quadratique moyenne en rajoutant une pondération qui permet de donner des poids différents aux différentes fréquences. Nous développons ensuite quatre méthodes de synthèse, l'utilisation de l'une ou de l'autre étant fonction du type de transformée utilisée (unitaire, non unitaire) et de la pondération des fréquences (pondéré ou non pondéré).

Cette généralisation permet de réduire la distorsion au détriment d'une augmentation du coût de filtrage. Un compromis entre la distorsion et le coût de filtrage devient alors nécessaire.

Les résultats de ce chapitre ont fait l'objet de la publication d'une communication à la conférence

internationale ICASSP-2009 [Daher2009-b].

Dans la suite de ce chapitre, nous utilisons les mêmes notations que celles définies dans le chapitre précédent. Le coût de calcul des différents algorithmes est estimé par le nombre de multiplications réelles.

3.2 Principe de la conception généralisée OLS/OLA

La structure matricielle du modèle classique de filtrage par blocs que nous avons développée dans le chapitre précédent montre que le filtre bloc est modélisé par une matrice G diagonale. Notre conception généralisée [Burel2004, Daher2009-b] qui sera développée dans ce chapitre consiste à considérer des matrices G quelconques, comme montré sur la figure 3.1. Les autres étapes de filtrage restent les mêmes. Dans le modèle classique, l'optimisation porte uniquement sur les éléments

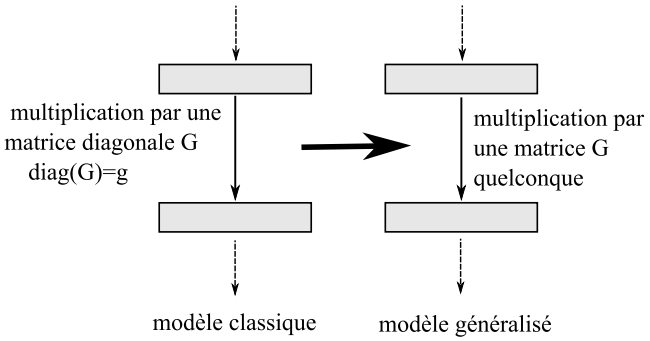

FIGURE 3.1 – Principe d'overlap-save et d'overlap-add généralisés

diagonaux de G étant donné que les autres coefficients sont nuls. Dans ce nouveau modèle généralisé, l'optimisation peut porter sur chacun des éléments de la matrice G.

3.3 Structures matricielles du filtrage overlap-save et overlap-add généralisé

Pour cette conception généralisée, le modèle de filtrage par blocs ainsi que celui du filtrage désiré restent les mêmes que ceux développés dans le chapitre précédent. Le principe de cette conception généralisée est alors décrit par les mêmes équations qui ont été développées dans le chapitre précédent et qui sont présentées ci-dessous. La différence réside dans la structure de la matrice G qui n'est plus forcément une matrice diagonale.

Pour l'overlap-save, le traitement d'un bloc d'entrée x_k, de taille M, fournit un bloc de sortie

y_k, de taille L, donné par :

$$y_k = A_s x_k \tag{3.1}$$

avec :

$$A_s = S F_M^{-1} G_s F_M \tag{3.2}$$

Le signal global de sortie y_s est alors donné par :

$$y_s = H_s x \tag{3.3}$$

H_s est la matrice carrée d'ordre K formée de b matrices blocs A_s, construite selon la structure indiquée sur la figure 2.2 du chapitre précédent. Le spectre $\overline{y_s}$ de ce signal y_s est alors donné par :

$$\overline{y_s} = \overline{\overline{H_s}}\,\overline{x} \tag{3.4}$$

où

$$\overline{\overline{H_s}} = F_K H_s F_K^{-1} \tag{3.5}$$

Pour l'overlap-add, le traitement d'un bloc d'entrée x_k, de taille L, fournit un bloc de sortie y_k, de taille M, donné par :

$$y_k = A_a x_k \tag{3.6}$$

avec

$$A_a = F_M^{-1} G_a F_M S^T \tag{3.7}$$

Le signal global de sortie y_a est alors donné par :

$$y_a = H_a x \tag{3.8}$$

H_a est la matrice carrée d'ordre K formée de b matrices blocs A_a construite selon la structure indiquée sur la figure 2.4 du chapitre précédent. Ainsi, le spectre $\overline{y_a}$ de ce signal y_a est donné par :

$$\overline{y_a} = \overline{\overline{H_a}}\,\overline{x} \tag{3.9}$$

où

$$\overline{\overline{H_a}} = F_K H_a F_K^{-1} \tag{3.10}$$

Dans la suite de ce chapitre, les notations en indice s et a respectivement relatives aux techniques d'OLS et d'OLA seront omises. Nous développons les algorithmes et les calculs en considérant uniquement le cas d'OLS. Des résultats analogues peuvent être obtenus pour le cas d'OLA.

3.4 Critère d'optimisation des filtres blocs

Le critère standard de mesure de la distance quadratique entre le spectre désiré et le spectre obtenu au moyen du filtrage par blocs permet d'évaluer la qualité du filtrage dans le cas où toutes les fréquences sont affectées de la même pondération. Cependant, dans certaines applications, il se trouve des cas où certaines fréquences ou bandes de fréquences doivent être favorisées par rapport aux autres. La qualité du filtrage est alors déterminé par une généralisation de ce critère de mesure qui consiste à pondérer les différentes fréquences. Souvent, la pondération utilisée est binaire où chaque fréquence est pondérée par 0 ou 1. La pondération zéro est affectée aux fréquences des bandes de garde du filtrage. Le cas classique, que l'on appellera par la suite critère non pondéré, consiste alors à pondérer toutes les fréquences à 1.

En tenant compte de cette pondération fréquentielle, l'optimisation du filtrage par blocs consiste donc à minimiser l'erreur quadratique moyenne pondérée suivante :

$$e_{QM} = \boldsymbol{E}\left\{\sum_{k=0}^{K-1} z\left(k\right)\left|\overline{y}\left(k\right) - \overline{y_d}\left(k\right)\right|^2\right\} = \boldsymbol{E}\left\{\left\|z \bullet \left(\overline{y} - \overline{y_d}\right)\right\|^2\right\} \tag{3.11}$$

Dans cette dernière équation, $\boldsymbol{E}$ désigne l'espérance mathématique, $|.|$ le module d'un complexe, $\bullet$ le produit terme à terme et $\|.\|$ la norme euclidienne d'un vecteur. z est le vecteur de taille K dont les composantes $z\left(k\right)$ correspondent aux coefficients de pondération affectés aux différentes fréquences.

Rappelons que pour le filtrage désiré, nous avons :

$$\overline{y_d} = \overline{\overline{H_d}}\overline{x} \tag{3.12}$$

$\overline{\overline{H_d}}$ est une matrice diagonale de taille K tel que $\overline{\overline{H_d}}\left(k,k\right) = f\left(k\right)$ où $f\left(k\right)$ est la réponse en fréquence

du filtre désiré. L'application de la transformée de Fourier discrète inverse permet d'écrire :

$$y_d = H_d x \tag{3.13}$$

où

$$H_d = F_K^{-1} \overline{\overline{H_d}} F_K \tag{3.14}$$

En notant par Z la matrice diagonale de taille K telle que $Z(k,k) = z(k)$, l'erreur (3.11) s'écrit :

$$e_{QM} = \boldsymbol{E}\left\{\left\|Z\left(\overline{y} - \overline{y_d}\right)\right\|^2\right\} \tag{3.15}$$

En remplaçant $\overline{y}$ et $\overline{y_d}$ par leurs expressions en fonction du spectre d'entrée $\overline{x}$ données respectivement par (3.4) et (3.12), nous obtenons :

$$e_{QM} = \boldsymbol{E}\left\{\left\|Z\left(\overline{\overline{H}}\overline{x} - \overline{\overline{H_d}}\overline{x}\right)\right\|^2\right\} = \boldsymbol{E}\left\{\left\|Z\left(\overline{\overline{H}} - \overline{\overline{H_d}}\right)\overline{x}\right\|^2\right\} \tag{3.16}$$

En supposant que le signal d'entrée x est un bruit blanc, donc de spectre uniforme, la minimisation de l'erreur e_{QM} revient à celle de l'erreur e telle que :

$$e = \left\|Z\left(\overline{\overline{H}} - \overline{\overline{H_d}}\right)\right\|_f^2 \tag{3.17}$$

$\|.\|_f$ désigne la norme de Frobenius d'une matrice.

La transformée le plus souvent utilisée en filtrage par blocs est la transformée de Fourier discrète (TFD) pour laquelle l'algorithme de calcul rapide par la transformée de Fourier rapide (TFR) est développé. Cependant, d'autres transformées admettant la propriété de la convolution cyclique peuvent être utilisées tel que la transformée en cosinus discrète (TCD) ou autres. D'autre part, les pondérations des différents fréquences peuvent être égales (critère non pondéré) ou différentes (critère pondéré). Pour concevoir le filtre bloc optimal, nous développerons quatre méthodes. Le domaine de validité de chaque méthode dépend du type de critère utilisé, pondéré ou non pondéré, et du type de la transformée utilisée, unitaire ou non unitaire. Dans le cas où plusieurs de ces méthodes sont applicables, elles fournissent le même résultat mais elles diffèrent par la complexité de calcul et le besoin d'espace mémoire.

3.5 Cas d'un critère pondéré : méthodes 1 et 2

Dans le cas d'un critère pondéré, la matrice Z est différente de la matrice unité. La norme de Frobenius d'une matrice étant égale à la norme euclidienne du vecteur formé par la juxtaposition de ses lignes, ainsi l'erreur donnée par l'équation (3.17) peut s'écrire :

$$e = \left\| ligne\left(Z\overline{\overline{H}}\right) - q_d \right\|^2 \tag{3.18}$$

où $ligne(Z\overline{\overline{H}})$ est le vecteur formé par la juxtaposition des lignes de la matrice $Z\overline{\overline{H}}$. q_d est le vecteur colonne de taille K^2 tel que $q_d = ligne\left(Z\overline{\overline{H_d}}\right)$.

En se basant sur la linéarité des calculs matriciels, un développement de calcul donné en *annexe* C permet d'écrire :

$$e = \|Fg - q_d\|^2 \tag{3.19}$$

où g est le vecteur colonne correspondant aux éléments libres de la matrice G. F est la matrice dont les colonnes de taille K^2 sont données selon la procédure décrite en *annexe C*.

3.5.1 Méthode 1

Cette méthode est valide pour n'importe quelle transformée utilisée. La solution optimale qui minimise l'erreur donnée par l'équation (3.19) est obtenue par la méthode de la pseudo-inverse :

$$g_{opt} = \left(F^\star F\right)^{-1} F^\star q_d \tag{3.20}$$

où $F^\star$ désigne la matrice adjointe, aussi appelée matrice transjuguée, de la matrice F.

Complexité de la méthode 1

Soit s le nombre d'éléments libres de la matrice G. Ainsi, le vecteur inconnu g est de taille s et la matrice F est de taille $K^2 \times s$.

La première étape de la méthode 1 consiste à calculer la matrice F et le vecteur q_d. En supposant que les coefficients des matrices diagonales Z et $\overline{\overline{H_d}}$ sont des réels, le calcul de q_d exige K multiplications réelles. D'autre part, le calcul de chaque colonne de F exige un nombre de multiplications réelles de l'ordre de $2K^2\left(log_2 K + 1\right)$ (*annexe C*).

La deuxième étape de cette méthode est le calcul de la pseudo-inverse selon l'équation (3.20). Le calcul le plus coûteux est celui de $F^\star F$ qui nécessite un nombre de multiplications complexes de l'ordre de s^2K^2, soit $3s^2K^2$ multiplications réelles. Dans le cas où la pseudo-inverse est calculée en utilisant la méthode de la décomposition en valeurs singulières (Singular value decomposition) [Golub1989], le coût de calcul de cette étape en globale est de l'ordre de $4s^2K^2$ multiplications réelles.

Au total, le coût de cette méthode est de l'ordre de $2sK^2\left(2s + log_2K + 1\right)$ multiplications réelles. Il est évident que cette méthode 1 exige un grand espace mémoire pour stocker la matrice F et un coût de calcul trop élevé même dans le cas des valeurs très réduites de s.

3.5.2 Méthode 2

Pour diminuer la complexité engendrée par le calcul de la matrice F et de la pseudo-inverse dans la méthode 1 ainsi que pour éviter la saturation de la mémoire provoquée par le stockage de la matrice F, nous proposons une autre méthode, appelée méthode 2, basée sur l'exploitation des propriétés de la TFD dans le cas où c'est celle-ci qui est utilisée. En effet, en calculant la matrice Ω_β donnée par :

$$\Omega_\beta = F_K^{-1}\left(H_{00}^\star \boxdot H_\beta\right) F_K \tag{3.21}$$

où $\boxdot$ désigne l'opérateur du produit d'Hadamard (multiplication terme à terme des matrices), et H_β est telle que $\overline{\overline{H_\beta}} = Z\overline{\overline{H_{m_\beta n_\beta}}}Z^\star$, les coefficients $F^\star F\left(\alpha,\beta\right)$, $\alpha,\beta = 0,1,...,s-1$, de la matrice carrée $F^\star F$ peuvent être obtenus par (*annexe D*) :

$$F^\star F\left(\alpha,\beta\right) = \Omega_\beta\left(l_\alpha, k_\alpha\right) \tag{3.22}$$

où $l_\alpha = \frac{Km_\alpha}{M}$ et $k_\alpha = \frac{Kn_\alpha}{M}$. m_α et n_α sont respectivement les indices de la ligne et de la colonne qui correspondent au $\alpha^{ème}$ élément libre de la matrice G, m_β et n_β sont respectivement les indices de la ligne et de la colonne qui correspondent au $\beta^{ème}$ élément libre de la matrice G.

Complexité de la méthode 2

Comme nous avons déjà signalé, la complexité du calcul de pseudo-inverse donné par l'équation (3.20) réside surtout dans le calcul de la matrice carrée $F^\star F$. Dans le cas où la matrice G possède s éléments libres, ce calcul de $F^\star F$ exige un calcul de s matrices Ω_β. Chaque matrice Ω_β exige un

nombre de multiplications réelles de l'ordre $4K^2log_2K + 7K^2$ (*annexe D*). En négligeant le coût des autres calculs, le coût de cette méthode est ainsi de l'ordre $2sK^2\left(2log_2K + \frac{7}{2}\right)$ multiplications réelles. Une réduction de complexité d'un facteur R_{12} de l'ordre de $\frac{(2s+log_2K+1)}{\left(2log_2K+\frac{7}{2}\right)}$ est ainsi obtenue par comparaison avec la méthode 1. Pour illustration, considérons une matrice G diagonale, ayant donc $s = M$ éléments libres, cas où $M = 2^9$ et $K = 2^{18}$, une réduction de complexité d'un facteur $R_{12} = 26.4$ est alors obtenue.

3.6 Cas d'un critère non pondéré : méthodes 3 et 4

Dans le cas où les fréquences ne sont pas pondérées, c'est-à-dire les coefficients de pondération $z(k)$ sont tous égaux à 1, la matrice diagonale de pondération Z est égale à la matrice identité I_K. L'erreur à minimiser définie par l'équation (3.17) peut donc s'écrire :

$$e = \left\| \left(\overline{\overline{H}} - \overline{\overline{H_d}} \right) \right\|_f^2 \tag{3.23}$$

Sachant que les matrices F_K et F_K^{-1} sont unitaires, la propriété de conservation de la norme de Frobenius permet d'écrire :

$$e = \|H - H_d\|_f^2 \tag{3.24}$$

où H et H_d sont données respectivement par les équations (3.5) et (3.14).

Comme décrit dans le chapitre précédent, la structure circulante de la matrice H_d et la structure particulière de la matrice H donnée par la figure 2.2 du chapitre précédent permettent de ramener la minimisation de l'erreur e à celle de l'erreur e_1 définie par :

$$e_1 = \|A - A_d\|_f^2 \tag{3.25}$$

A_d étant la matrice extraite de la matrice H_d de la même manière que A est extraite de la matrice H.

En remplaçant A par son expression donnée par la relation (3.2), cette erreur e_1 peut s'écrire :

$$e_1 = \left\| ST_M^{-1} G T_M - A_d \right\|_f^2 \tag{3.26}$$

Cette erreur e_1 a la même forme que celle obtenue dans le chapitre 2. Cependant, dans ce chapitre,

la matrice G n'est plus nécessairement diagonale et la transformée T_M n'est plus nécessairement la TFD.

3.6.1 Méthode 3

Cette méthode peut être appliquée quelle que soit la transformée T_M utilisée. La norme de Frobenius d'une matrice étant égale à la norme du vecteur formé par la juxtaposition de ses lignes, l'erreur e_1 de l'équation (3.25) peut s'écrire :

$$e_1 = \|ligne\,(A) - a_d\|^2$$

a_d étant le vecteur colonne de taille LM tel que $a_d = ligne(A_d)$.

D'après l'*annexe C*, cette erreur s'écrit :

$$e_1 = \|Eg - a_d\|^2 \tag{3.27}$$

où g est le vecteur colonne correspondant aux éléments libres de la matrice G. E est la matrice dont les colonnes de taille LM sont données selon la procédure décrite dans l'*annexe C*.

La solution optimale est alors obtenue par un calcul de pseudo-inverse :

$$g_{opt} = (E^\star E)^{-1} E^\star a_d \tag{3.28}$$

Complexité de la méthode 3

Soit s le nombre d'éléments libres de la matrice G. Ainsi, le vecteur inconnu g est de taille s et la matrice E est de taille $LM \times s$.

La première étape de cette méthode consiste à calculer la matrice E et le vecteur a_d. Le calcul du vecteur a_d nécessite le calcul de la matrice A_d extraite de la matrice H_d obtenue par un calcul de TFD de la diagonale de $\overline{\overline{H_d}}$, soit un nombre de multiplications réelles de l'ordre de $Klog_2K$. Le coût de calcul des colonnes de la matrice E dépend de la transformée T_M utilisée mais peut être négligé si $M \ll K$ (*annexe C*).

Le calcul le plus complexe à réaliser dans la pseudo-inverse donnée en (3.28) est celui de $E^\star E$. Celui-ci exige un nombre de multiplications complexes de l'ordre de s^2LM, soit $3s^2LM$ multiplications réelles. Dans le cas où la pseudo-inverse est calculée en utilisant la méthode de la décomposition

en valeurs singulières (Singular value decomposition) [Golub1989], le coût de calcul de cette étape en global est de l'ordre de $4s^2LM$ multiplications réelles.

Au total, le coût de cette méthode est de l'ordre de $Klog_2K + 4s^2LM$ multiplications réelles.

3.6.2 Méthode 4

Cette méthode ne peut être utilisée que dans le cas où la transformée utilisée est unitaire.

La transformée utilisée étant unitaire, la multiplication par les matrices unitaires T_M et T_M^{-1} ne modifie pas la norme de Frobenius. Par conséquent, e_1 peut s'écrire :

$$e_1 = \left\| ST_M^{-1}G - A_dT_M^{-1} \right\|_f^2 \tag{3.29}$$

soit,

$$e_1 = \|BG - C\|_f^2 \tag{3.30}$$

où B et C sont les matrices $L \times M$ données par $B = ST_M^{-1}$ et $C = A_dT_M^{-1}$

Dans le cas où aucune relation n'existe entre des coefficients les différentes colonnes de G, ces colonnes peuvent être déterminées séparément. En effet, l'erreur e_1 peut s'écrire :

$$e_1 = \sum_{m=0}^{M-1} \|BG(:,m) - C(:,m)\|^2 \tag{3.31}$$

où $G(:,m)$ et $C(:,m)$ représentent respectivement les colonnes des matrices G et C.

Chaque colonne $G(:,m)$ peut alors être obtenue en minimisant l'erreur partielle donnée par :

$$e_{1,m} = \|BG(:,m) - C(:,m)\|^2 \tag{3.32}$$

Dans le cas où les éléments d'une colonne $G(:,m)$ ne sont pas tous libres, le système peut être réduit en supprimant les colonnes de la matrice B correspondant aux éléments nuls du vecteur $G(:,m)$.

La solution optimale minimisant chaque erreur partielle $e_{1,m}$ est ainsi obtenue par la méthode de la pseudo-inverse, soit :

$$G(:,m)_{opt} = \left(B^\star B\right)^{-1} B^\star C(:,m) \tag{3.33}$$

Dans le cas d'une matrice G diagonale, cette méthode donne le même résultat que celui obtenu par la méthode rapide développée dans le chapitre 2.

Complexité de la méthode 4

Soit s le nombre d'éléments libres de la matrice G. Nous supposons que les s éléments libres de la matrice G sont uniformément répartis entre les colonnes, c'est-à-dire chaque colonne contient s/M éléments libres. Les matrices B définies ci-dessous sont alors de taille $L \times \frac{s}{M}$. Ainsi, cette méthode exige les calculs suivants :

- Calcul de la matrice $B = ST_M^{-1}$ qui consiste à sélectionner L lignes de la matrice T_M^{-1}. Ce calcul n'exige aucune multiplication.
- Calcul de la matrice $C = A_d T_M^{-1}$ qui consiste à calculer tout d'abord la matrice A_d et à appliquer ensuite une transformée inverse sur ses lignes. La matrice A_d est extraite de la matrice H_d dont le calcul exige une TFD $K-$points de la diagonale de la matrice $\overline{\overline{H_d}}$, soit un nombre de multiplications réelles de l'ordre de $Klog_2K$. Le calcul de $C = A_d T_M^{-1}$ exige un nombre de multiplications réelles de l'ordre de $LMlog_2M$.
- Calcul des colonnes $G(:,m)_{opt}$: le calcul le plus complexe est celui de $B^\star B$. Chaque colonne exige $\left(\frac{s}{M}\right)^2 L$ multiplications complexes, soit $3\frac{s^2}{M}L$ multiplications réelles pour les M colonnes de G.

Au total, le coût de cette méthode est de l'ordre de $\left(Klog_2K + LMlog_2M + \frac{3s^2L}{M}\right)$ multiplications réelles.

3.7 Résumé des différentes méthodes

Le tableau 3.1 résume le domaine de validité de chaque méthode ainsi que l'ordre du coût de synthèse du filtre optimal exprimé en nombre de multiplications réelles. s est le nombre d'éléments libres de la matrice G. La dernière colonne de ce tableau représente le coût de calcul pour un cas typique : $L = M/2$ et $K = 4M$ et $s = M$.

Pour déterminer la méthode la moins complexe en fonction de la transformée utilisée et la pondération choisie, un récapitulatif est donnée sur la figure 3.2. Dans le cas où la transformée utilisée est une TFD, la méthode 2 peut être utilisée pour le critère pondéré et la méthode 4 pour un critère non pondéré.

TABLE 3.1 – Coût de calcul des quatre méthodes

Méthode	Transformée utilisée	Critère de pondération	Ordre du coût de calcul	Coût typique pour $L = M/2$, $K = 4M$ et $s = M$
1	n'importe	n'importe	$\left(2sK^2\left(2s + log_2K + 1\right)\right)$	$32M^3\left(2M + log_2M + 3\right)$
2	TFD	n'importe	$\left(2sK^2\left(2log_2K + \frac{7}{2}\right)\right)$	$32M^3\left(2log_2M + \frac{15}{2}\right)$
3	n'importe	non pondéré	$\left(Klog_2K + 4s^2LM\right)$	$2M^3\left(M + \frac{2(log_2M+2)}{M^2}\right)$
4	Unitaire	non pondéré	$\left(Klog_2K + LMlog_2M + \frac{3s^2L}{M}\right)$	$\frac{M^2}{2}\left(log_2M + 3 + \frac{8(log_2M+2)}{M}\right)$

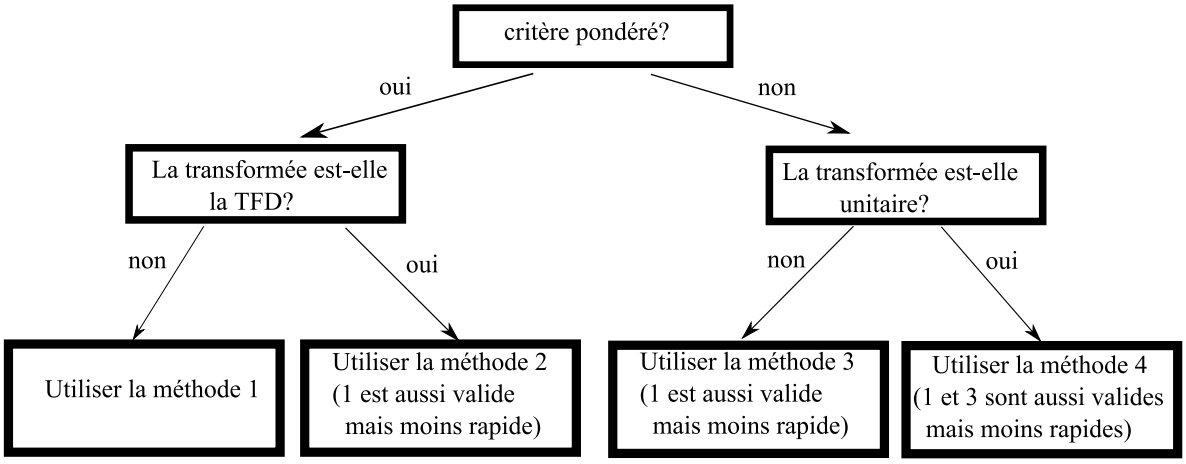

FIGURE 3.2 – Schéma récapitulatif pour le choix de la méthode de synthèse la mieux adaptée

3.8 Coût du filtrage par blocs généralisé

Considérons le cas ou la transformée utilisée est la TFD. Pour chaque bloc d'entrée de taille M, le filtrage selon le modèle de la conception généralisée du filtrage par blocs exige :

- $\mathcal{O}\left(Mlog_2M\right)$ multiplications réelles pour le calcul de la TFD directe.
- s multiplications complexes pour la multiplication du vecteur obtenu par la matrice G ayant s éléments libres, soit $3s$ multiplications réelles.
- $\mathcal{O}\left(Mlog_2M\right)$ multiplications réelles pour le calcul de la TFD inverse.

Au total, un nombre de multiplications réelles de l'ordre de $3s + 2Mlog_2M$ est nécessaire pour le traitement de chaque bloc, soit $\frac{3s+2Mlog_2M}{L}$ multiplications réelles par échantillon de sortie. Dans le cas d'une matrice G diagonale, le rapport de complexité entre le filtrage par blocs selon la conception généralisée et le filtrage par blocs selon la conception classique est donné par :

$$R = \frac{3s + 2Mlog_2M}{M\left(3 + 2log_2M\right)}$$

soit,

$$R = \frac{1 + \frac{3}{2log_2 M}\left(\frac{s}{M}\right)}{1 + \frac{3}{2log_2 M}}$$

Notons que, dans le cas où les blocs sont traités directement par un calcul $y_k = Ax_k$ selon la figure 3.3 sans passage par les étapes intermédiaires du calcul de la TFD, chaque bloc de sortie de L échantillons exige LM multiplications complexes, soit $3LM$ multiplications réelles. Par conséquent, l'utilisation de la TFD permet de réduire le coût de calcul dans le cas où :

$$3s + 2Mlog_2 M < 3LM \tag{3.34}$$

soit,

$$s < M\left(L - \frac{2}{3}log_2 M\right) \tag{3.35}$$

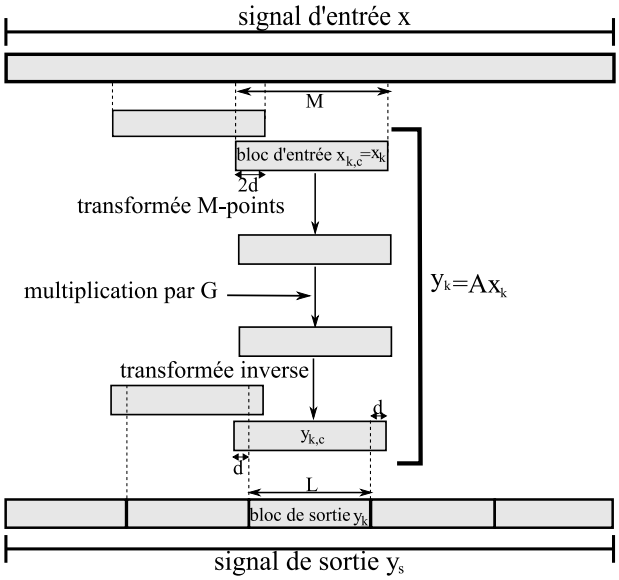

FIGURE 3.3 – Schéma d'overlap-save non causal

3.9 Distorsion du filtre bloc

Le filtrage par blocs étant de type LPVT de période L, l'erreur de distorsion totale e, donnée par $e = \left\| \left(\overline{\overline{H}} - \overline{\overline{H_d}}\right) \right\|_f^2$, est décomposée en deux parties (voir chapitre précédent, section 2.7) :

- la distorsion LIT : $e_i = \sum_{k=0}^{K-1} \left| \overline{\overline{H}}(k,k) - \overline{\overline{H_d}}(k,k) \right|^2$;
- le repliement de spectre (aliasing) : $e_r = \sum_{k=0}^{K-1} \left(\sum_{l=0, l\neq k}^{K-1} \left| \overline{\overline{H}}(k,l) - \overline{\overline{H_d}}(k,l) \right|^2 \right)$.

Cette erreur exprimée aussi par $e = \|(H - H_d)\|_f^2$, peut être décomposée en deux parties :

- e_{indep} qui ne dépend pas du choix des coefficients de G ;
- e_{dep} qui représente la partie de l'erreur qui dépend du choix des coefficients de G et peut s'écrire en fonction de l'erreur e_1 définie dans l'équation (3.25) :

$$e_{dep} = be_1 \tag{3.36}$$

où $b = K/L$ est le nombre de blocs x_i du signal x.

Nous pouvons ainsi écrire :

$$e = e_i + e_r = e_{dep} + e_{indep} \tag{3.37}$$

L'erreur e_{indep} ne dépend que du choix des tailles M et L des blocs d'entrée et de sortie que nous supposons constants dans notre étude.

3.10 Représentation des résultats

Nous présentons et interprétons les résultats obtenus à partir des mêmes hypothèses que celles définies dans le chapitre 2. Les tailles des blocs sont données par $M = 32$ et $L = 24$ et la résolution fréquentielle correspond à $K = 96$. Le filtre désiré est un filtre passe bas dont la représentation fréquentielle est indiquée sur la figure 2.5 du chapitre précédent.

3.10.1 Conception optimale : critère pondéré - critère non pondéré

Pour évaluer notre approche de conception optimale en cas de pondération des fréquences, nous comparons les résultats de la conception sans pondération et les résultats de la conception avec pondération dans le cas où la transformée utilisée est la TFD. Pour simplifier, nous avons considéré le cas où les éléments libres de la matrice G sont ceux de sa diagonale principale. Les coefficients de pondération sont tous égaux à 1 à l'exception de ceux qui correspondent aux bandes de transition définies entre $k = 8$ et $k = 13$ d'un part et entre $k = 83$ et $k = 88$ de l'autre part. Les coefficients du vecteur de pondération z sont représentés sur la figure 3.4. La méthode la moins complexe pour concevoir le filtre optimal en cas de pondération est la méthode 2. Dans le cas du critère non pondéré, la méthode la moins complexe est la méthode 4, ou encore, la méthode rapide développée dans le chapitre précédent qui fournit les mêmes résultats. Les figures 3.5, 3.6 et 3.7 montrent respectivement les erreurs de distorsion LIT, les erreurs de repliement de spectre et les erreurs globales obtenues

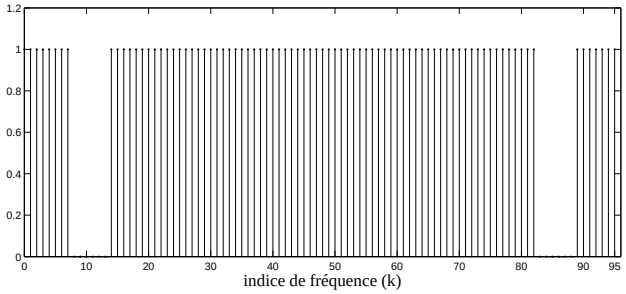

FIGURE 3.4 – Coefficients de pondération des fréquences

dans les deux cas. Ces résultats montrent que la pondération des fréquences permet de réduire les erreurs de repliement de spectre pour toutes les fréquences. Les erreurs LIT sont réduites dans les bandes passantes et de coupure alors qu'elles augmentent dans les bandes de garde. Globalement, la pondération des fréquences a permis de réduire l'erreur totale dans la bande passante et la bande de coupure où les fréquences sont pondérées à 1 en prenant bénéfice des bandes de transition où les fréquences sont pondérées à 0.

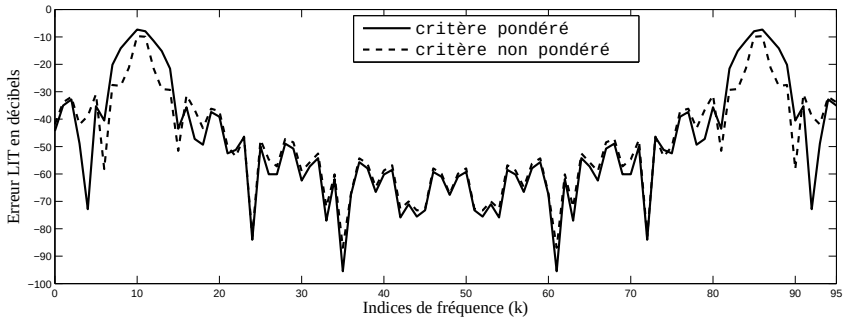

FIGURE 3.5 – Comparaison des erreurs LIT : critère pondéré, critère non pondéré

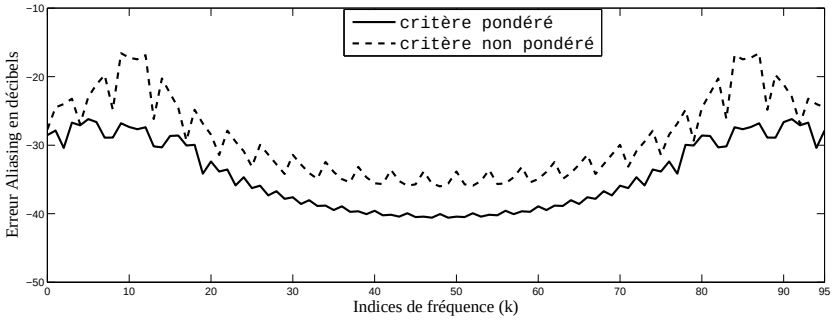

FIGURE 3.6 – Comparaison des erreurs de repliement de spectre : critère pondéré, critère non pondéré

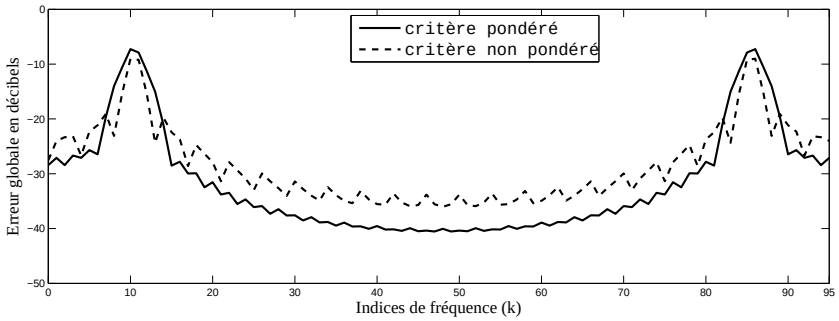

FIGURE 3.7 – Comparaison des erreurs totales : critère pondéré, critère non pondéré

3.10.2 Conception optimale : G non diagonale - G diagonale

Nous considérons le cas où la transformée utilisée est la TFD et le critère est non pondéré. Pour évaluer notre approche de conception généralisée dans les cas des matrices G non diagonales, nous avons considéré le cas particulier où toutes les lignes et les colonnes de la matrice G contiennent le même nombre n d'éléments libres. Pour chaque ligne i, $i = 0, 1, ..., M-1$, nous supposons que les éléments libres sont ceux qui correspondent à $G(i, \langle i+\alpha \rangle_M)$, $\alpha = -\left[\frac{n+1}{2}\right] + 1, ...0, 1, ..., \left[\frac{n}{2}\right]$, où [.] désigne la partie entière d'un réel positif.

La méthode la moins complexe pour concevoir le filtre optimal dans ce cas est la méthode 3. Les simulations sont réalisées pour un nombre d'éléments libres n allant de 1 à 15. Ces résultats de simulations, représentés par la figure 3.8, montre que l'erreur indépendante e_{indep} reste constante puisqu'elle ne dépend pas de la matrice G. L'erreur dépendante e_{dep}, et par conséquent l'erreur globale, diminuent au fur et à mesure que le nombre des éléments libres croit.

Ainsi, cette généralisation de la structure de la matrice G permet une réduction de l'erreur de distorsion au prix d'une augmentation dans le coût de filtrage et le coût de synthèse du filtre. Le tableau 3.2 permet de comparer les valeurs des différentes erreurs obtenues dans le cas d'une matrice G diagonale et dans le cas d'une matrice G à 3 éléments libres dans chaque ligne/colonne (G tri-diagonale).

TABLE 3.2 – Comparaison des erreurs de distorsion en fonction de la structure de G

Méthode optimale	Time-invariant	Aliasing	e_{indep}	e_{dep}	Distorsion totale
G diagonale	0.4647	0.3291	0.7329	0.0609	0.7938
G tri-diagonale	0.4661	0.2971	0.7329	0.0302	0.7632

Ces résultats fournis par le tableau 3.2 montrent que l'erreur dépendante et l'erreur globale sont moins importantes dans le cas où G est tri-diagonale que dans le cas où G est diagonale. En effet,

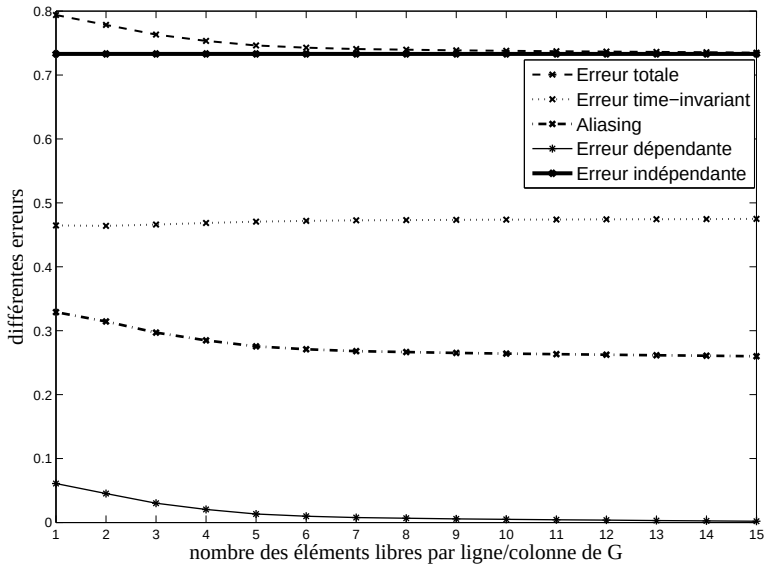

FIGURE 3.8 – Distorsion du filtre optimal en fonction du nombre des éléments libres de G

dans le cas où G est diagonale, l'optimisation du filtrage par minimisation de l'erreur e_1 définie par l'équation (3.26) porte uniquement sur les éléments diagonaux de la matrice G. Cependant, dans le cas où G est tri-diagonale, l'optimisation s'applique aussi aux autres éléments libres.

3.11 Conclusion

Dans ce chapitre, une conception généralisée des techniques de filtrage OLS et OLA a été introduite. Quatre méthodes de conception optimale des filtres blocs ont été proposées et développées. Le choix d'une de ces méthodes dépend du type de pondération des fréquences et du type de transformée utilisée. Ces méthodes se différencient par leur coût d'implémentation et par l'espace mémoire exigé. Des résultats de simulation et des comparaisons ont été ensuite exposés. Dans le cas d'un critère non pondéré et utilisant une transformée unitaire, la TFD en particulier, la méthode 4 peut être appliquée. Dans le cas classique d'une matrice G diagonale, le filtre optimal peut être obtenu par la méthode de synthèse rapide qui a été développée dans le chapitre 2.

L'étude théorique relative aux erreurs, développée dans les chapitres 2 et 3, reste valable en pratique pour une implémentation en virgule flottante. Cependant, pour une implémentation en virgule fixe, des erreurs supplémentaires liées à la quantification des termes de la TFD et à l'arrondissement des calculs intermédiaires viennent s'ajouter à l'erreur de distorsion du filtre.

Pour éliminer les effets de calcul en virgule fixe, nous proposons dans les chapitres suivants une

implémentation d'OLS et d'OLA mettant en oeuvre la transformée en nombres de Fermat (FNT : Fermat Number Transform). Cette transformée, décrite dans le chapitre suivant, permet d'un côté une large réduction du nombre de multiplications nécessaires à la réalisation du produit de convolution circulaire et de l'autre côté un calcul sans erreur d'arrondi dans le cas d'une implémentation en virgule fixe.

Chapitre 4

Les transformées en nombres entiers

4.1 Introduction

L'implantation temps réel des algorithmes de traitement du signal, en particulier du filtrage, est soumise à un certain nombre de contraintes telles que le temps d'exécution et la facilité de mise en oeuvre. Les processeurs disponibles sur le marché se regroupent en deux grandes catégories : les DSP à virgule flottante, pour lesquels les calculs sont relativement faciles à mettre en oeuvre mais présentent toutefois des inconvénients de consommation et de prix, et les processeurs à virgule fixe qui répondent mieux aux contraintes technologiques et économiques mais pour lesquels l'implantation d'algorithmes est plus délicate à réaliser [Frantz2004, Andraka2004]. L'absence d'unité arithmétique en nombre flottant rend les DSP à virgule fixe meilleur marché tout en permettant une grande vitesse de traitement des données. Pour surmonter certains difficultés de programmation sur ce dernier type de DSP, nous allons introduire un outil mathématique, appelé transformée en nombres entiers (NTT : Number Theoretic Transform), largement appréciable pour les calculs de convolution cyclique en virgule fixe, dans le but d'optimiser l'implantation du filtrage par blocs sur des processeurs à virgule fixe.

Initialement développées pour permettre un calcul rapide de la convolution, les transformées en nombres entiers restent, jusqu'à nos jours, très peu utilisées dans le domaine du traitement du signal. Cela est dû à la domination de l'algorithme de transformée de Fourier rapide et au développement de la technologie qui nous fournit des DSP à virgule flottante avec une rapidité de calcul largement suffisante pour nos applications de base. Cependant, pour les applications complexes, les DSPs à virgule fixe sont largement utilisés et l'outil de la NTT est intéressant pour réduire la complexité

des algorithmes dans le domaine du traitement du signal tel que les fonctions de filtrage. En effet, comparée à la transformée de Fourier discrète ou toute autre transformée, la transformée en nombres entiers présente les avantages suivants :

- Tous les calculs s'effectuent sur l'ensemble des entiers relatifs $\mathbb{Z}$, ce qui permet une précision de calcul améliorée pour une implantation sur un processeur DSP à virgule fixe. Les calculs dans $\mathbb{Z}$ étant exacts, toute erreur d'arrondi est supprimée [Julien1991].
- L'utilisation de la transformée en nombres entiers évite le passage en complexe ou en réel inhérent à la transformée de Fourier discrète ou autres transformées, ce qui est particulièrement intéressant en terme de coûts de calcul et de performances. En effet, une multiplication complexe requiert au moins trois multiplications réelles alors qu'une multiplication dans $\mathbb{Z}$ est une opération simplifiée.

Dans ce chapitre, nous introduisons la notion de transformée en nombres entiers (NTT : Number Theoretic Transform) et en particulier celle de la transformée en nombres de Fermat (FNT : Fermat Number Transform). La connaissance des opérations arithmétiques avec les calculs effectués en congruence sur des corps de Galois étant nécessaire, plusieurs résultats et outils connus du domaine de la théorie des nombres entiers sont rappelés. Pour conclure, une comparaison entre la transformée en nombres de Fermat et la transformée de Fourier discrète permet de présenter les principaux avantages de la mise en oeuvre de la FNT dans l'implantation des méthodes OLS et OLA sur des processeurs DSP à virgule fixe.

4.2 Rappels arithmétiques

4.2.1 Calcul de congruences

Propriété 4.1

Soit x et y deux éléments de l'ensemble $\mathbb{Z}$ des entiers relatifs et q est un entier naturel. x et y sont dits congrus modulo q, notation $\langle x \equiv y \rangle_q$, si et seulement si x et y ont le même reste de la division euclidienne par q, ce qui est équivalent à $x - y$ multiple de q.

$$\langle x \equiv y \rangle_q \Longleftrightarrow x - y = kq \quad k \in \mathbb{Z} \tag{4.1}$$

Exemple : En notant par $x \, mod \, q$ le reste de la division euclidienne d'un élément x par q , nous avons $21 \, mod \, 17 = 4 \, mod \, 17 \Longleftrightarrow \langle 21 \equiv 4 \rangle_{17}$.

Propriété 4.2

Pour un entier naturel q, la congruence modulo q donne une relation d'équivalence dans $\mathbb{Z}$. Les classes d'équivalence seront données par l'ensemble $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z} = \{0, 1, 2, ..., q-1\}$. En effet, quel que soit x de $\mathbb{Z}$, x peut s'écrire sous la forme $x = kq + r$ où r est le reste de la division euclidienne de x par q et l'on a $0 \leq r \leq q-1$.

Propriété 4.3

$$\langle a+b \rangle_q = \left\langle \langle a \rangle_q + \langle b \rangle_q \right\rangle_q \tag{4.2}$$

$$\langle a.b \rangle_q = \left\langle \langle a \rangle_q . \langle b \rangle_q \right\rangle_q \tag{4.3}$$

$$\left\langle a^b \right\rangle_q = \left\langle \left(\langle a \rangle_q \right)^b \right\rangle_q \tag{4.4}$$

Définition 4.1

L'inverse a^{-1} d'un élément a appartenant à $\mathbb{Z}_q$ est défini comme étant le plus petit entier positif qui vérifie $\left\langle aa^{-1} \equiv 1 \right\rangle_q$

Propriété 4.4

L'inverse b d'un élément a appartenant à $\mathbb{Z}_q$ existe dans $\mathbb{Z}_q$ si et seulement si a et q sont premiers entre eux :

$$a\, est\, inversible\, modulo\, q \iff pgcd(a, q) = 1 \tag{4.5}$$

Conséquence 4.1

Avec des opérations arithmétiques + et × définis dans $\mathbb{Z}_q$ (calcul modulo q), nous avons :

- Si q n'est pas un nombre premier, il existe alors des éléments non nuls de $\mathbb{Z}_q$ qui ne possèdent pas d'inverses appartenant à $\mathbb{Z}_q$. $(\mathbb{Z}_q, +, \times)$ est dans ce cas un anneau fini.
- Si q est un nombre premier, tout élément non nul de $\mathbb{Z}_q$ possédera un inverse. $(\mathbb{Z}_q, +, \times)$ est alors un corps fini. Ce corps d'ordre q est appelé corps de Galois et noté $GF(q)$ (voir *annexe E*).

Exemple : L'ensemble $GF(2) = \{0, 1\}$ doté des opérations arithmétiques + et × en modulo (1+1 = 0) est un corps de Galois utilisé pour la représentation binaire des nombres.

Corollaire 4.1

Pour tout entier a et nombre premier p , $a^{p-1}-1$ est un multiple de p.

Théorème 4.1

Le petit théorème de Fermat : Si a est un entier non divisible par p tel que p est un nombre premier, alors $\left(a^{p-1}-1\right)$ est un multiple de p, donc $\left\langle a^{p-1} \equiv 1\right\rangle_p$

Définition 4.2 : La fonction indicatrice d'Euler

La fonction d'Euler $\varphi(n)$ est la fonction qui associe à chaque entier n le nombre d'entiers positifs inférieurs à n et qui sont premiers avec n. Euler a introduit cette fonction $\varphi(n)$ pour formuler la généralisation du petit théorème de Fermat.

Exemple : $\varphi(8) = 4$ car les nombres premiers avec 8 et inférieurs à 8 sont au nombre de quatre : 1, 3, 5 et 7.

Lemme 4.1

Si p est un nombre premier alors : $\varphi(p) = p-1$

Lemme 4.2

Si p est un nombre premier et α un entier ≥ 1 alors : $\varphi(p^\alpha) = p^{\alpha-1}(p-1)$

Lemme 4.3

Si m et n sont premiers entre eux alors : $\varphi(mn) = \varphi(m)\varphi(n)$

Théorème 4.2

Soit a et q deux entiers premiers entre eux, alors $\left\langle a^{\varphi(q)} \equiv 1\right\rangle_q$

4.2.2 Détermination de l'inverse

Il existe plusieurs méthodes pour calculer l'inverse d'un élément sur le corps de Galois $GF(q)$. La méthode la plus utilisée est celle de l'algorithme d'Euclide qui reste valable quel que soit l'entier a à inverser et l'ordre q du corps de Galois.

Proposition 4.1 *Le calcul de l'inverse d'un entier a dans le corps de Galois GF(q) consiste à résoudre l'équation* :

$$\langle ax + kq \equiv 1 \rangle_q \tag{4.6}$$

où $k \in \mathbb{Z}$ *et* x *représente un entier inconnu égal à l'inverse de* a *modulo* q.

La solution de cette équation peut être obtenue par l'algorithme d'Euclide étendue (voir *annexe F*). D'autres méthodes existent pour déterminer un inverse mais qui ne sont utilisables que pour des conditions très particulières. Ainsi pour les valeurs de q égales à un nombre de Fermat défini par $q = 2^{2^t} + 1$ avec $t \in \mathbb{N}$, des calculs plus efficaces d'inverses existent.

Proposition 4.2 *L'inverse d'un entier a appartenant à GF (q) avec q un nombre premier de Fermat,* $q = 2^{2^t} + 1$ *et* $t \in \mathbb{N}$, *est donné par :*

$$\left\langle a^{-1} = a^{2^{2^t}-1} a^{2^{2^t}-2} ... a^{2^1} a^{2^0} \right\rangle_q \tag{4.7}$$

Démonstration

D'après le petit théorème de Fermat, la congruence $\left\langle x^{q-1} \equiv 1 \right\rangle_q$ est facilement vérifiable pour tout entier $x \in \mathbb{Z}_q$ tel que q est un nombre premier. L'équation (4.7) peut alors être démontrée de la manière suivante :

$$\left\langle a^{-1} \equiv a^{-1}.1 \equiv a^{-1}a^{q-1} \equiv a^{q-2} \equiv a^{2^{2^t}-1} \right\rangle_q$$

– Pour $t = 0$, nous avons :

$$\left\langle a^{-1} \equiv a^{2^{2^t}-1} \equiv a^{2^1-2^0} \equiv a^{2^0} \right\rangle_q$$

– Pour $t = 1$, nous obtenons la congruence suivante :

$$\left\langle a^{-1} \equiv a^{2^{2^t}-1} \equiv a^{2^2-2^0} \equiv \frac{a^{2^2}}{a^{2^0}} \equiv \frac{a^{2^1}a^{2^1}}{a^{2^0}} \equiv a^{2^1}a^{2^0} \right\rangle_q$$

– Pour $t > 1$, nous déduisons :

$$\begin{cases} \left\langle a^{-1} \equiv a^{2^{2^t}-1} \equiv a^{2^{2^t}-2^0} \equiv \frac{a^{2^{2^t}-1}a^{2^{2^t}-1}}{a^{2^0}} \equiv \frac{a^{2^{2^t}-1}a^{2^{2^t}-2}a^{2^{2^t}-2}}{a^{2^0}} \right\rangle_q \\ \left\langle a^{-1} \equiv \frac{a^{2^{2^t}-1}a^{2^{2^t}-2}...a^{2^2}}{a^{2^0}} \equiv \frac{a^{2^{2^t}-1}a^{2^{2^t}-2}...a^{2^1}a^{2^1}}{a^{2^0}} \right\rangle_q \\ \left\langle a^{-1} \equiv a^{2^{2^t}-1}a^{2^{2^t}-2}...a^{2^1}a^{2^0} \right\rangle_q \end{cases}$$

Ce calcul permet de déterminer l'inverse d'un entier dans $GF(q)$ où $q = 2^{2^t} + 1$ et exige $2(2^{2^t} - 1)$ multiplications.

Une autre méthode, plus contraignante, qui n'est valable que pour des entiers a appartenant à $GF\left(2^{2^t} + 1\right)$ et égaux à une puissance de 2, ce qui assure une valeur entière à $log_2(a)$, est présentée dans la proposition ci-dessous.

Proposition 4.3 *Pour tout entier $a = 2^k$ avec $k \in \mathbb{N}^*$, l'inverse de a modulo $q = 2^{2^t} + 1$ peut être obtenu par :*

$$\left\langle a^{-1} \equiv -2^{2^t - k} \equiv -2^{2^t - log_2(a)} \right\rangle_{2^{2^t}+1} \tag{4.8}$$

Démonstration

A partir de la congruence $\left\langle 2^{2^t} \equiv -1 \right\rangle_{2^{2^t}+1}$, l'équation (4.8) peut être démontrée ainsi :

$$\left\langle 2^{2^t} 2^{-k} \equiv -1.2^{-k} \right\rangle_q$$

$$\left\langle 2^{-k} \equiv -2^{2^t}.2^{-k} \right\rangle_q$$

L'inverse de $a = 2^k$ est alors donné par :

$$\left\langle a^{-1} = 2^{-k} \equiv -2^{2^t - k} \right\rangle_q$$

Dans le cas d'implémentation numérique binaire, ce calcul d'inverse ne requiert pas de multiplication mais uniquement de décalage de bits. Cette dernière proposition sera utilisée par la suite pour le calcul de la transformée inverse en nombres de Fermat.

4.3 Transformées en nombres entiers

Les transformées en nombres entiers ont été développées pour permettre un calcul rapide et sans erreur d'arrondi des produits de convolution des séquences entières dans les applications numériques. Une transformée en nombres entiers (NTT) présente la même forme que celle de la transformée de Fourier discrète (TFD) [Julien1991]. Cependant, son domaine de définition n'est plus l'ensemble des complexes $\mathbb{C}$ mais un corps de Galois $GF(q)$ d'ordre q premier ou pseudo-premier.

La transformée en nombres entiers d'une séquence $\{x_n\}_{n=0}^{M-1}$ de taille M dont les éléments x_n appartiennent au corps de Galois $GF(q)$ est une séquence $\{X_k\}_{k=0}^{M-1}$ de même taille M. Les éléments

X_k appartiennent au même corps de Galois $GF(q)$ et sont déterminés par :

$$X_k = \left\langle \sum_{n=0}^{M-1} x_n \alpha^{nk} \right\rangle_q \quad k = 0, 1, ..., M-1 \tag{4.9}$$

α représente le terme générateur d'ordre M dans le corps de Galois $GF(q)$:

$$\left\langle \alpha^M \equiv 1 \right\rangle_q \tag{4.10}$$

Si la longueur M de la transformée et le modulo q sont premiers entre eux, il existe un inverse M^{-1} dans le corps de Galois $GF(q)$ tel que $\left\langle M.M^{-1} \equiv 1 \right\rangle_q$. La transformée inverse de la NTT existe alors et elle est donnée par :

$$x_n = \left\langle M^{-1} \sum_{n=0}^{M-1} X_k \alpha^{-nk} \right\rangle_q \quad n = 0, 1, ..., M-1 \tag{4.11}$$

Pour qu'une transformée en nombres entiers et son inverse existent, les paramètres α, M et q doivent vérifier les conditions suivantes [Agarwal1974] :

- Condition d'existence de α^{-1} et de M^{-1} :

$$pgcd\,(\alpha, q) = pgcd\,(M, q) = 1 \tag{4.12}$$

- Si q est premier, M doit diviser $(q-1)$:

$$M \mid (q-1) \tag{4.13}$$

  En effet, d'après le petit théorème de Fermat, nous pouvons écrire $\left\langle \alpha^{q-1} \equiv 1 \right\rangle_q$. En effectuant la division euclidienne de $(q-1)$ par M, nous obtenons :

$$\left\langle \alpha^{mM+r} \equiv 1 \right\rangle_q$$

  où r est le reste de cette division tel que $0 \leq r \leq M-1$. Soit,

$$\left\langle \left(\alpha^M\right)^m \alpha^r \equiv 1 \right\rangle_q$$

$$\langle \alpha^r \equiv 1 \rangle_q$$

α étant d'ordre M (M est le plus petit entier différent de zéro tel que $\langle \alpha^M \equiv 1 \rangle_q$), nous devons donc avoir $r = 0$, c'est-à-dire que M divise $(q-1)$.

– Si q n'est pas premier, il peut s'écrire sous la forme d'un produit de puissance m_i de nombres p_i premiers entre eux $q = \prod p_i^{m_i}$. Dans ce cas, la longueur M devra diviser le plus grand commun diviseur des nombres $p_i - 1$:

$$M | pgcd(p_i - 1, p_j - 1) \ \forall \ i \neq j \tag{4.14}$$

En effet, les p_i étant premiers entre eux, la relation $\langle \alpha^M \equiv 1 \rangle_q$ implique que :

$$\begin{cases} \langle \alpha^M \equiv 1 \rangle_{p_1^{m_1}} \\ \langle \alpha^M \equiv 1 \rangle_{p_2^{m_2}} \\ \quad \vdots \end{cases}$$

Par conséquent, d'après le théorème d'Euler, nous obtenons :

$$M | \varphi\left(p_i^{m_i}\right)$$

soit,

$$M | p_i^{m_i - 1}\left(p_i - 1\right)$$

p_i étant un nombre premier, nous obtenons :

$$M | \left(p_i - 1\right)$$

4.3.1 Différents types de transformées en nombres entiers

D'une façon générale, d'après la définition de la NTT et de son inverse, le calcul d'une transformée en nombres entiers de longueur M exige M^2 multiplications et $M(M-1)$ additions. Ces multiplications par des puissances de la racine $n^{ème}$ de l'unité α constituent l'opération arithmétique la plus complexe rencontrée dans le calcul de ces transformées. L'efficacité de l'implantation d'une NTT sur un processeur DSP et plus encore d'une intégration VLSI (Very Large Scale Integration)

dépend en grande partie de l'optimisation de cette opération de multiplication. Par exemple, pour une valeur de α puissance de 2, ces multiplications par des puissances de deux sont remplacées par des simples décalages de bits, ce qui diminue énormément le coût de calcul d'une NTT et par conséquent de tout algorithme utilisant cette transformée.

Le premier papier apparu dans ce domaine était celui de *Rader* [Rader1972] dans lequel sont développées les transformées en nombres de Mersenne (MNT : Mersenne Number Transform) définies sur un ensemble d'entiers d'ordre égal à un nombre de Mersenne $q = 2^p - 1$ avec p un entier premier. Dans ce cas, les opérations arithmétiques en modulo q peuvent être effectuées au moyen de l'opération 'complément à un' et sont ainsi facilement réalisables. Cependant, sachant que la longueur M de la transformée doit diviser $q-1$, le choix des longueurs M reste relativement limité du fait de la faible décomposition possible de $q-1$ dans le cas où q est un nombre de Mersenne.

Plus tard, *Agarwal* et *Burrus* [Agarwal1974] ont montré que pour certaines longueurs de séquences, la transformée en nombres entiers définie sur un corps d'ordre égal à un nombre de Fermat peut être mise en oeuvre à l'aide d'additions, de soustractions et de décalage de bits. Une étude comparative des transformées en nombres entiers et le choix de l'ordre q a été présentée dans [Agarwal1975].

Par la suite, plusieurs autres types de transformées en nombres entiers ont été développées [Nussbaumer1976, Nussbaumer1977, Nussbaumer1978, Chevillat1978, Blahut1985]. La transformée en nombres de Fermat reste la plus adaptée dans la mesure où elle permet :

- une exécution simple des opérations en modulo q.
- un choix de α puissance de 2 permettant ainsi de réaliser les multiplications avec des simples décalages de bits [Duhamel1982].
- La possibilité d'une implémentation par une structure Butterfly équivalente à celle de la TFR.

Avant de détailler la transformée en nombres de Fermat qui nous servira dans le chapitre suivant, nous donnons quelques définitions et propriétés liées aux transformées en nombres entiers et au corps de Galois $GF(q)$.

4.3.2 Propriétés des transformées en nombres entiers

Propriété 4.5

Pour un terme générateur α d'ordre M de $GF(q)$, tous les produits nk exposants de α dans l'équation (4.9) *de la transformée directe et tous les produits $-nk$ dans l'équation* (4.11) *de la trans-*

formée inverse peuvent être calculés en modulo M :

$$\left\langle \alpha^{nk} \equiv \alpha^{\langle nk \rangle_M} \right\rangle_q$$

$$\left\langle \alpha^{-nk} \equiv \alpha^{\langle -nk \rangle_M} \right\rangle_q$$

Démonstration

La division euclidienne de nk par M permet d'écrire :

$$nk = aM + \langle nk \rangle_M , \quad a \in \mathbb{Z}$$

Ainsi,

$$\begin{cases} \left\langle \alpha^{nk} \right\rangle_q &= \left\langle \alpha^{aM+\langle nk \rangle_M} \right\rangle_q \\ &= \left\langle \alpha^{aM} . \alpha^{\langle nk \rangle_M} \right\rangle_q \\ &= \left\langle \left\langle \alpha^{aM} \right\rangle_q . \left\langle \alpha^{\langle nk \rangle_M} \right\rangle_q \right\rangle_q \\ &= \left\langle \left\langle \left(\left\langle \alpha^{M} \right\rangle_q \right)^a \right\rangle_q . \left\langle \alpha^{\langle nk \rangle_M} \right\rangle_q \right\rangle_q \\ &= \left\langle 1 . \left\langle \alpha^{\langle nk \rangle_M} \right\rangle_q \right\rangle_q \\ &= \left\langle \alpha^{\langle nk \rangle_M} \right\rangle_q \end{cases}$$

L'ensemble des propriétés associées à la TFD existent aussi pour les transformées en nombres entiers. En considérant une séquence $\{x_n\}$ et sa transformée en nombres entiers $\{X_k\}$, ces différentes propriétés sont énoncés ci-dessous d'une manière non-exhaustive [Agarwal1974].

Propriété 4.6 : linéarité

La transformée en nombres entiers de la somme de deux séquences $\{x_1\}$ et $\{x_2\}$ est égale à la somme sur $GF(q)$ de leurs transformées :

$$NTT(x_1 + x_2) = \langle NTT(x_1) + NTT(x_2) \rangle_q$$

où T représente une transformation en nombres entiers directe ou inverse.

Propriété 4.7 : périodicité

Si la séquence $\{x_n\}$ est périodique de période m, $x_n = x_{n+m}$, alors sa transformée en nombres entiers $\{X_k\}$ est aussi périodique de même période, c'est-à-dire $X_k = X_{k+m}$.

Propriété 4.8 : *symétrie et antisymétrie*

- *Si la séquence* $\{x_n\}$ *est symétrique* ($x_{m-n} = x_{m+n}$) *alors sa transformée en nombres entiers est aussi symétrique* ($X_{m-k} = X_{m+k}$)
- *Si la séquence* $\{x_n\}$ *est antisymétrique* ($x_{m-n} = -x_{m+n}$) *alors sa transformée en nombres entiers est aussi antisymétrique* ($X_{m-k} = -X_{m+k}$)

Propriété 4.9 : *décalage*

Si la séquence $\{x_n\}$ *est soumise à un décalage de* m *échantillons alors sa transformée en nombres entiers est donnée par :*

$$NTT\left(\{x_{n+m}\}\right) = NTT\left(\{x_n\}\right)\alpha^{-mk} = \left\{X_k\alpha^{-mk}\right\}$$

Propriété 4.10 : *convolution cyclique*

La séquence $\{y_n\}$ de taille M issue de la convolution cyclique de deux séquences de nombres entiers $\{x_n\}$ et $\{h_n\}$ également de taille M s'obtient par :

$$\{y\left(n\right)\} = \{x_n\} * \{h_n\} = INTT\left(NTT\left(\{x_n\}\right) \bullet NTT\left(\{h_n\}\right)\right) \tag{4.15}$$

où l'opérateur $\bullet$ désigne la multiplication terme à terme.

Notons que la relation de congruence $\langle -a \equiv q - a \rangle_q$, valable dans un ensemble d'entiers $\mathbb{Z}$ d'ordre q, indique que tout entier négatif $(-a)$ est représenté par un entier positif. Ainsi, pour éviter une double interprétation des entiers $y\left(n\right)$ résultant du calcul de convolution cyclique donné par l'équation (4.15), les composantes $y\left(n\right)$ doivent respecter la condition $\mid y\left(n\right) \mid \leq \frac{q}{2}$. Or, les valeurs des éléments $y\left(n\right)$ sont bornés par :

$$\mid y\left(n\right) \mid_{max} \leq M \mid x\left(n\right) \mid_{max} \mid h\left(n\right) \mid_{max}$$

Ainsi, une condition suffisante pour pallier à tout dépassement *'overflow'* dans le calcul de $\{y\left(n\right)\}$ est donnée par :

$$M \mid x\left(n\right) \mid_{max} \mid h\left(n\right) \mid_{max} \leq \frac{q}{2}$$

soit,

$$\mid x\left(n\right) \mid_{max} \mid h\left(n\right) \mid_{max} \leq \frac{q}{2M} \tag{4.16}$$

Si les valeurs de l'une des séquences sont connues, $h(n)$ par exemple, les bornes des valeurs de $y(n)$ pourront être affinées telles que :

$$\mid y(n) \mid_{max} \leq \mid x(n) \mid_{max} \sum_{n=0}^{M-1} \mid h(n) \mid$$

Dans ce cas, une condition suffisante pour pallier à tout dépassement *'overflow'* dans le calcul de $\{y(n)\}_{n=0}^{M-1}$ est donnée par :

$$\mid x(n) \mid_{max} \sum_{n=0}^{M-1} \mid h(n) \mid \leq \frac{q}{2}$$

soit,

$$|x(n)|_{max} \leq \frac{q}{2 \sum_{n=0}^{M-1} \mid h(n) \mid} \tag{4.17}$$

4.4 Transformée en nombres de Fermat (FNT)

Les transformées en nombres de Fermat (FNT : Fermat Number Transform) sont définies dans un ensemble $GF(q)$ dont l'ordre est égal à un nombre de Fermat $q = F_t = 2^{2^t} + 1$ avec t un entier [Julien1991]. Notons que seuls les cinq premiers nombres de Fermat sont premiers :

$$\begin{cases} F_0 = 2^{2^0} + 1 = 2^1 + 1 = 3 \\ F_1 = 2^{2^1} + 1 = 2^2 + 1 = 5 \\ F_2 = 2^{2^2} + 1 = 2^4 + 1 = 17 \\ F_3 = 2^{2^3} + 1 = 2^8 + 1 = 257 \\ F_4 = 2^{2^4} + 1 = 2^{16} + 1 = 65537 \end{cases}$$

Les suivants, étant considérés comme pseudo-premiers, pourront être utilisés pour le calcul des transformées [Rosen1993]. Cependant le choix des longueurs M dans ce cas est limité par la condition d'existence de la transformée et de son inverse (M à diviser le plus grand diviseur commun de leurs facteurs premiers diminués de l'unité) :

$$\begin{cases} F_5 = 2^{2^5} + 1 = 2^{32} + 1 = 4294967297 = 641 \times 6700417 \\ F_6 = 2^{2^6} + 1 = 2^{64} + 1 = 274177 \times 67280421310721 \end{cases}$$

Une transformée en nombres de Fermat de longueur M puissance de 2 peut être implémentée en

utilisant la même structure Butterfly mise en oeuvre pour la TFR, ce qui exige de l'ordre de $\frac{M}{2} log_2 M$ multiplications et de $M log_2 M$ additions en modulo F_t. Le terme générateur α étant essentiel pour réduire la complexité, la FNT permet un choix de α égal à une puissance de deux. Ce choix très intéressant en arithmétique binaire permet de réaliser toutes les multiplications par les puissances de α au moyen des décalages de bits. Les longueurs possibles M de transformées et les termes générateurs α associés sont respectivement donnés par :

$$M = 2^{t+1-i} \tag{4.18}$$

$$\left\langle \alpha \equiv 2^{2^i} \right\rangle_{F_t} \tag{4.19}$$

avec i et t des entiers tels que $0 \leq i < t$. La longueur de la séquence $M = 2^{t+1-i}$ étant égale à une puissance de 2, son inverse est donnée par :

$$\left\langle M^{-1} = -2^{2^t - t + i - 1} \right\rangle_{F_t} \tag{4.20}$$

En effet :

$$\left\langle F_t - 1 = 2^{2^t} \equiv -1 \right\rangle_{F_t}$$

$$\left\langle 2^{2^t} \left(-2^{-(t-i+1)}\right) \equiv (-1)\left(-2^{-(t-i+1)}\right) \right\rangle_{F_t}$$

$$\left\langle -2^{2^t - t + i - 1} = 2^{-t+i-1} = M^{-1} \right\rangle_{F_t}$$

Agarwal et *Burrus* [Agarwal1974] ont montré qu'il est possible, en conservant une valeur de α égale à une puissance de 2, de doubler la longueur de la transformée M en posant $i = -1$ (voir (4.1)). Le terme générateur est alors défini comme étant la solution de la congruence suivante :

$$\left\langle \alpha^2 \equiv 2 \right\rangle_{F_t}$$

La valeur de α telle que $\left\langle \alpha^2 \equiv 2 \right\rangle_{F_t}$ existe pour tout nombre de Fermat F_t avec $t \geq 2$ et peut être exprimée par :

$$\left\langle \alpha \equiv 2^{2^{t-2}} \left(2^{2^{t-1}} - 1\right) \right\rangle_{F_t} \tag{4.21}$$

En effet,

$$\alpha^2 = 2 = (-2)(-1)$$

$$\left\langle \alpha^2 = (-2)\left(2^{2^t}\right) = 2^{2^{t-1}}\left(-1 - 2\left(2^{2^{t-1}}\right) + 1\right)\right\rangle_{F_t}$$

$$\left\langle \alpha^2 = \left(2^{2^{t-2}}\right)^2\left(\left(2^{2^{t-1}}\right)^2 - 2\left(2^{2^{t-1}}\right) + 1\right)\right\rangle_{F_t}$$

$$\left\langle \alpha^2 = \left(2^{2^{t-2}}\left(2^{2^{t-1}} - 1\right)\right)^2\right\rangle_{F_t}$$

d'où la relation $\left\langle \alpha \equiv \sqrt{2} \equiv 2^{2^{t-2}}\left(2^{2^{t-1}} - 1\right)\right\rangle_{F_t}$.

La notation $\alpha = \sqrt{2}$ souvent utilisée pour la simplification des écritures désigne α appartenant à $GF(F_t)$ tel que $\langle \alpha^2 \equiv 2\rangle_{F_t}$. Les multiplications par une puissance n de $\sqrt{2}$ peuvent être réalisées par un décalage de bits si n est pair ou par deux décalages et une addition si n est impair. En effet, pour $\alpha = \sqrt{2}$, nous avons :

$$\left\{ \begin{array}{lcll} \left\langle \left(\sqrt{2}\right)^n \right\rangle_{F_t} & = & \left\langle 2^{n/2} \right\rangle_{F_t} & si\ n\ est\ pair \\ \left\langle \left(\sqrt{2}\right)^n \right\rangle_{F_t} & = & \left\langle \left(\sqrt{2}\right)^{n-1}\sqrt{2} \equiv 2^{\frac{(n-1)}{2}}\left(2^{2^{t-2}}\left(2^{2^{t-1}} - 1\right)\right)\right\rangle_{F_t} & si\ n\ est\ impair \end{array} \right\}$$

TABLE 4.1 – Paramètres possibles pour l'implémentation de la FNT

t	b	modulo F_t	M pour $\alpha = 4$	M pour $\alpha = 2$	M pour $\alpha = \sqrt{2}$	M_{max}	α pour M_{max}
0	1	$2^1 + 1$	–	2	–	2	2
1	2	$2^2 + 1$	2	4	–	2^2	2 ou 3
2	4	$2^4 + 1$	4	8	16	2^4	$\sqrt{2}$ ou 3
3	8	$2^8 + 1$	8	16	32	2^8	3
4	16	$2^{16} + 1$	16	32	64	2^{16}	3
5	32	$2^{32} + 1$	32	64	128	2^7	$\sqrt{2}$
6	64	$2^{64} + 1$	64	128	256	2^8	$\sqrt{2}$

Agarwal et *Burrus* [Agarwal1974] ont vérifié que la transformée en nombres de Fermat admet la propriété de convolution cyclique et qu'un calcul de FNT requiert environ $Mlog_2M$ opérations de décalages de bits et d'additions mais aucune multiplication, alors qu'une transformée de Fourier discrète exige un nombre de multiplications réelles de l'ordre de $\left(\frac{M}{2}log_2\frac{M}{2}\right)$ [Malvar1992].

Pour illustrer l'utilisation de la propriété de la convolution cyclique de la transformée en nombres de Fermat (FNT), un exemple est traité pour le calcul de convolution de deux séquences x et h de longueur $M = 4$: $x = (2, -2, 1, 0)$, $h = (1, 2, 0, 0)$.

En considérant le choix $t = 2$ donc $F_2 = 17$, la racine $\alpha = 4$ répond à l'égalité $\langle 4^4 \equiv 1 \rangle_{17}$. Sachant que $\langle \alpha^M \equiv 1 \rangle_q$, donc $\langle \alpha^{nk} \equiv \alpha^{\langle nk \rangle_M} \rangle_q$, la matrice T associée à la transformation directe s'écrit alors :

$$T = \left\langle \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 4 & 4^2 & 4^3 \\ 1 & 4^2 & 4^4 & 4^6 \\ 1 & 4^3 & 4^6 & 4^9 \end{pmatrix} \right\rangle_{17} = \left\langle \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 4 & 4^2 & 4^3 \\ 1 & 4^2 & 1 & 4^2 \\ 1 & 4^3 & 4^2 & 4^1 \end{pmatrix} \right\rangle_{17}$$

soit,

$$T = \left\langle \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 4 & 16 & 13 \\ 1 & 16 & 1 & 16 \\ 1 & 13 & 16 & 4 \end{pmatrix} \right\rangle_{17}$$

L'inverse de $M = 4$ dans $GF(17)$ étant égal à $\langle M^{-1} = 4^{-1} = 13 \rangle_{17}$, la matrice T^{-1} associée à la transformation inverse s'écrit :

$$T^{-1} = \left\langle 13 \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 4^{-1} & 4^{-2} & 4^{-3} \\ 1 & 4^{-2} & 4^{-4} & 4^{-6} \\ 1 & 4^{-3} & 4^{-6} & 4^{-9} \end{pmatrix} \right\rangle_{17} = \left\langle 13 \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 4^3 & 4^2 & 4^1 \\ 1 & 4^2 & 1 & 4^2 \\ 1 & 4^1 & 4^2 & 4^3 \end{pmatrix} \right\rangle_{17}$$

soit,

$$T^{-1} = \left\langle \begin{pmatrix} 13 & 13 & 13 & 13 \\ 13 & 16 & 4 & 1 \\ 13 & 4 & 13 & 4 \\ 13 & 1 & 4 & 16 \end{pmatrix} \right\rangle_{17}$$

Les transformées en nombres de Fermat de h et x sont données respectivement par $\overline{x} = \langle T.x \rangle_{17}$ et $\overline{h} = \langle T.h \rangle_{17}$, soient,

$$\overline{x} = \left\langle \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 4 & 16 & 13 \\ 1 & 16 & 1 & 16 \\ 1 & 13 & 16 & 4 \end{pmatrix} \begin{pmatrix} 2 \\ -2 \\ 1 \\ 0 \end{pmatrix} \right\rangle_{17} = \left\langle \begin{pmatrix} 1 \\ 10 \\ 5 \\ 9 \end{pmatrix} \right\rangle_{17}$$

$$\overline{h} = \left\langle \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 4 & 16 & 13 \\ 1 & 16 & 1 & 16 \\ 1 & 13 & 16 & 4 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \\ 0 \\ 0 \end{pmatrix} \right\rangle_{17} = \left\langle \begin{pmatrix} 3 \\ 9 \\ 16 \\ 10 \end{pmatrix} \right\rangle_{17}$$

Le produit terme à terme de $\overline{h}$ par $\overline{x}$ donne :

$$\overline{y} = \left\langle \overline{x} \bullet \overline{h} \right\rangle_{17} = \left\langle \begin{pmatrix} 1 \\ 10 \\ 5 \\ 9 \end{pmatrix} \bullet \begin{pmatrix} 3 \\ 9 \\ 16 \\ 10 \end{pmatrix} \right\rangle_{17} = \left\langle \begin{pmatrix} 3 \\ 5 \\ 12 \\ 5 \end{pmatrix} \right\rangle_{17}$$

La convolution y est alors obtenue par une transformation inverse en nombres de Fermat de $\overline{y}$:

$$y = T^{-1}\overline{y}$$

$$y = \left\langle \begin{pmatrix} 13 & 13 & 13 & 13 \\ 13 & 16 & 4 & 1 \\ 13 & 4 & 13 & 4 \\ 13 & 1 & 4 & 16 \end{pmatrix} \begin{pmatrix} 3 \\ 5 \\ 12 \\ 5 \end{pmatrix} \right\rangle_{17} = \left\langle \begin{pmatrix} 2 \\ 2 \\ 14 \\ 2 \end{pmatrix} \right\rangle_{17} = \begin{pmatrix} 2 \\ 2 \\ -3 \\ 2 \end{pmatrix}$$

Ce résultat est bien identique à celui obtenu par le calcul de convolution conventionnel donné par :

$$y_k = \sum_{n=0}^{3} h(n)\, x(\langle k-n \rangle_4) \quad k = 0, 1, 2, 3$$

Cet exemple illustre bien que les approximations telles que les troncatures ou les arrondis de nombres n'existent pas dans une arithmétique en nombres entiers. De plus, les séquences de départ satisfaisant la condition exprimée par l'équation (4.17), les différents dépassements observés durant les étapes intermédiaires de calcul n'ont pas d'effet sur le résultat final. Notons qu'une généralisation de la borne d'amplitude $\pm A$ des éléments de chaque séquence à convoluer, pour un modulo égal à un nombre de Fermat $F_t = 2^{2^t} + 1$ et une longueur de séquence $M = 2^b$, est donnée par $A \leq 2^{\frac{2^t - b - 1}{2}}$ [Alfredsson1996].

4.5 Mise en oeuvre d'une FNT

Pour le chapitre suivant, notre étude portera sur la mise en oeuvre du filtrage par bloc par le biais des transformées en nombres de Fermat (FNT). Le choix de la FNT s'explique par sa grande modularité et sa relative simplicité de réalisation. En effet, les propriétés de la FNT permettent d'envisager une exécution VLSI (Very Large Scale Integration) des opérations arithmétiques réalisables à l'aide des circuits logiques binaires.

De nombreuses manières de représenter les nombres entiers de l'ensemble $\mathbb{Z}_{2^b+1}$ par des mots binaires de $b+1$ bits existent. La complexité des opérations arithmétiques en calcul modulo et la performance des architectures dépendent de la représentation choisie. Une description de l'implantation des opérations arithmétiques (addition, soustraction et multiplication par une puissance de 2) sur le corps de Galois lorsque les données sont codées en complément à 2 est donnée dans [Agarwal1974]. Les représentations les plus connues sont celles proposées par *McClellan* [McClellan1976] et *Leibowitz* [Leibowitz1976]. En utilisant ces représentations, les opérations telles que l'addition et la multiplication par deux, peuvent être assez facilement effectuées en $VLSI$.

4.5.1 Implémentation des opérations binaires

Dans cette partie, nous présentons la mise en oeuvre des opérations d'addition (soustraction) et de multiplication réduites par un modulo q de la forme $q = 2^b + 1$ par le biais des fonctions binaires simples. Pour envisager l'utilisation d'un processeur b bits, les éléments appartenant à l'ensemble $\mathbb{Z}_{2^b+1}$ sont représentés par des mots binaires d'une longueur égale à b bits. Un bit supplémentaire sera nécessaire pour représenter 2^b ce qui compliquera l'intégration matérielle (*hardware*) sur une carte DSP. Ayant une probabilité d'apparition très faible (de l'ordre de 2^{-b}) pour des données décorrélées, la valeur 2^b ne pouvant pas être représentée sur b bits est remplacé par 0 ou $2^b - 1$ [Agarwal1974]. Les faibles erreurs occasionnées devront être bien entendu préalablement jugées acceptables en fonction du type d'application mise en oeuvre.

Une représentation binaire dans $\mathbb{Z}_{2^b+1}$ des opérations de réduction par le modulo $2^b + 1$, de négation, d'addition ou de multiplications sont détaillées ci-après. Notons que certaines de ces opérations seront bien évidemment impliquées dans le calcul des transformées en nombres de Fermat directe et inverse. Pour la suite de cette partie, posons un nombre entier $u = (u_{b-1}\, u_{b-2}\, \ldots\, u_1\, u_0)_2$ représenté par un mot de b bits tel que $u = \sum_{n=0}^{b-1} u_n 2^n$ où $u_n \in \mathbb{Z}_2 = \{0, 1\}$. Dans un souci de

clarté, les opérations binaires seront à chaque fois illustrées dans $\mathbb{Z}_{2^4+1}$, avec des entiers codées sur 4 bits ; l'extension des résultats au cas général de $\mathbb{Z}_{2^b+1}$ est évidente.

La réduction modulo $2^b + 1$

Toute opération dans un ensemble $\mathbb{Z}_{2^b+1}$ est suivie d'une réduction par le modulo $q = 2^b + 1$. Il est donc important que cette procédure soit la plus simple et rapide possible. Pour certaines opérations, ce calcul modulo pourra même être intégré dans le calcul.

Proposition 4.6 *Sachant que* $\left\langle 2^b \equiv -1 \right\rangle_{2^b+1}$, *le résidu modulo* $2^b + 1$ *d'un entier* $u \geq 2^b$, *écrit sur* $(b+1)$ *bits, est obtenu en retranchant la retenue* $u_b = 1$ *au mot* $u = (u_{b-1}\, u_{b-2}\, \ldots\, u_1\, u_0)_2$.

$$\left\langle u = u_b 2^b + \sum_{n=0}^{b-1} u_n 2^n = \sum_{n=0}^{b-1} u_n 2^n - u_b 2^0 = \sum_{n=0}^{b-1} u_n 2^n - 1 \right\rangle_{2^b+1}$$

Cette soustraction de 1 est réalisée par l'addition du complément à deux de 1 :

$$\left\langle u = \sum_{n=0}^{b-1} u_n 2^n + \sum_{n=0}^{b-1} 1.2^n \right\rangle_{2^b+1}$$

$$\left\langle u = \sum_{n=0}^{b-1} (u_n + 1)\, 2^n \right\rangle_{2^b+1}$$

Comme indiqué précédemment, pour le cas de $u = 2^b$, nous pouvons choisir d'arrondir u à 0 ou bien à $2^b - 1$.

Exemple : Prenons l'exemple de l'entier $u = 26$ à réduire par un modulo égal au deuxième nombre de Fermat $2^{2^2} + 1 = 17$ $(b = 4)$. La représentation binaire de l'entier $u = 26$, sur $b + 1 = 5$ bits, est donnée par :

$$26 = (11010)_2$$

La réduction modulo 17 de cet entier est réalisée par :

$$\langle 26 = (11010 - 0001)_2 = (11010 + 1111)_2 = (1001)_2 = 9 \rangle_{17}$$

La négation

Pour effectuer l'opération de changement de signe dans un ensemble $\mathbb{Z}_{2^b+1}$, la démarche est basée sur l'écriture en mode binaire de tout nombre entier u négatif sous la forme :

$$-u = -\sum_{n=0}^{b-1} u_n 2^n = \sum_{n=0}^{b-1} \overline{u_n} 2^n - \left(2^b - 1\right)$$

où $\overline{u_n}$ représente le complément à 1 du bit u_n.

Proposition 4.7 *Sachant que* $\left\langle 2^b \equiv -1 \right\rangle_{2^b+1}$, *l'opération de négation de tout entier, appartenant à un ensemble d'entiers d'ordre égal à* $2^b + 1$, *peut être réalisée par l'équation suivante* :

$$\left\langle -u = -\sum_{n=0}^{b-1} u_n 2^n = \sum_{n=0}^{b-1} \overline{u_n} 2^n - \left(2^b - 1\right) = \sum_{n=0}^{b-1} \overline{u_n} 2^n + 2 \right\rangle_{2^b+1}$$

Exemple : L'inverse de $u = 4$ est donné par :

$$\left\langle -4 = \sum_{n=0}^{b-1} \overline{u_n} 2^n + 2 = (1011 + 0010)_2 = (1101)_2 = 13 \right\rangle_{17}$$

L'addition et la soustraction

Considérons maintenant l'addition $\langle u + v \rangle_{2^b+1}$ où les entiers u et v, écrits sur b bits, appartiennent à $\mathbb{Z}_{2^b+1}$, soit $0 \leq u,\, v < 2^b$:

$$w = \langle u + v \rangle_{2^b+1} = \left\langle \sum_{n=0}^{b-1} u_n 2^n + \sum_{n=0}^{b-1} v_n 2^n \right\rangle_{2^b+1}$$

Proposition 4.8 *Si* $w < 2^b$, *alors* w *est obtenu simplement par l'équation* :

$$w = \sum_{n=0}^{b-1} u_n 2^n + \sum_{n=0}^{b-1} v_n 2^n$$

Sinon l'addition sera suivie d'une réduction modulo $2^b + 1$ *telle que* :

$$w = \left\langle \sum_{n=0}^{b-1} \left(u_n + v_n\right) 2^n - 1 \right\rangle_{2^b+1}$$

Proposition 4.9 *La soustraction* $\langle u - v \rangle_{2^b+1}$ *sera, elle, effectuée en deux temps. La négation*

de v est calculée puis additionnée à u.

La multiplication par une puissance de 2

Comme il a été déjà précisé précédemment, les multiplications par des puissances de deux sont particulièrement faciles à mettre en oeuvre en arithmétique binaire. En effet, la multiplication par 2 d'un nombre binaire $u = \sum_{n=0}^{b-1} u_n 2^n$ est réalisé par un décalage de bit vers la gauche, en ne tenant compte que des $(b-1)$ premiers bits :

$$2u = (u_{b-2}u_b, ...u_1u_00)_2$$

Proposition 4.10 *La multiplication par une puissance β de 2 dans l'ensemble $\mathbb{Z}_{2^b+1}$, avec $0 \leq 2^\beta u \leq 2^b - 1$, pourra donc être obtenue par β décalages de bits sur u :*

$$\left\langle u = \sum_{n=\beta}^{b-1} u_{(n-\beta)} 2^n \right\rangle_{2^b+1}$$

Si $2^\beta u \geq 2^b$, alors la procédure de réduction modulo $2^b + 1$ doit être effectuée. La retenue 1 sera alors soustraite pour maintenir l'opération à l'ensemble $\mathbb{Z}_{2^b+1}$.

Notons que le calcul d'une transformée inverse peut demander des multiplications du type $2^{-\beta}$. Dans ce cas, la puissance négative sera remplacée par la congruence $\left\langle 2^{-\beta} \equiv -2^{b-\beta} \right\rangle_{2^b+1}$. Un décalage de $b - \beta$ bits suivi d'une négation du résultat intermédiaire sont alors nécessaires pour exécuter l'opération.

Exemple : Deux possibilités existent pour le calcul binaire $\left\langle 2^\beta u \right\rangle_{2^b+1}$. Prenons par exemple l'opération $\left\langle 2^3 (11) = 3 \right\rangle_{17}$, avec $u = 11$ et $\beta = 3$, qui peut être calculée par 2 méthodes :

– Soit chaque décalage est dissocié et la réduction par le modulo est effectuée dès que nécessaire :

$$\langle 2 \times 11 \equiv 2 (1011)_2 \equiv (10110)_2 \equiv (0110)_2 + (1111)_2 \equiv (0101)_2 \rangle_{17}$$

$$\left\langle 2^2 \times 11 \equiv 2 (2 \times 11) \equiv 2 (0101) \equiv (1010)_2 \right\rangle_{17}$$

$$\left\langle 2^3 \times 11 \equiv 2 \left(2^2 \times 11\right) \equiv 2 (1010)_2 \equiv (10100)_2 \equiv (0100)_2 + (1111)_2 \equiv (0011)_2 \equiv 3 \right\rangle_{17}$$

– Soit les décalages de $\beta = 3$ bits à gauche sont pris en compte de manière globale tel que :

$$\left\langle 11 \times 2^3 \equiv 2^3 \times (1011)_2 \equiv (1011000)_2 \equiv (1000)_2 + (1011)_2 \equiv (0011)_2 \equiv 3 \right\rangle_{17}$$

La multiplication

Pour optimiser les opérations de multiplications, un produit standard de deux entiers quelconques $(u,\, v) \in \mathbb{Z}^2_{2^b+1}$ peut être exécuté comme une sommation de sous-produits.

Proposition 4.11 *L'opération de multiplication exécute* $\langle w = uv \rangle_{2^b+1}$ *avec* $u = \sum_{n=0}^{b-1} u_n 2^n$ *et* $v = \sum_{n=0}^{b-1} v_n 2^n$ *où* $(u_n,\, v_n) \in \mathbb{Z}^2_2$ *tel que* :

$$\langle w = uv \rangle_{2^b+1} = \left\langle \sum_{n=0}^{b-1} u_n \left(2^n v\right) = \sum_{n=0}^{b-1} u_n \left(2^n \sum_{m=0}^{b-1} v_m 2^m \right) \right\rangle_{2^b+1}$$

Si $w \geq 2^b$, *alors la procédure de réduction modulo* $2^b + 1$ *doit bien sûr être effectuée.*

Exemple : Prenons l'exemple d'une multiplication simple telle que $\langle w = 9 \times 13 = 15 \rangle_{17}$:

$$\begin{cases} \left\langle 9 \times 13 = (1001)_2 \times (1101)_2 = 2^0 (1101)_2 + 2^3 (1101)_2 = (1101)_2 + (1101000)_2 \right\rangle_{17} \\ \langle 9 \times 13 = (1101)_2 + (1000)_2 + (1010)_2 \rangle_{17} \\ \langle 9 \times 13 = (1101)_2 + (0010)_2 = (1111)_2 = 15 \rangle_{17} \end{cases}$$

Le calcul de puissance

Pour calculer la puissance $\langle w = u^v \rangle_{2^b+1}$, avec $u = \sum_{n=0}^{b-1} u_n 2^n$ et $v = \sum_{n=0}^{r-1} v_n 2^n$ où $(u_n,\, v_n) \in \mathbb{Z}^2_2$, la méthode la plus utilisée est appelée *binary method* [Knuth1969] et permet d'écrire w tel que :

$$\left\langle w = u^v = \left(\left(\left(\left(u^{v_{r-1}} \right)^2 u^{v_{r-2}} \right)^2 \ldots u^{v_2} \right)^2 u^{v_1} \right)^2 u^{v_0} \right\rangle_{2^b+1}$$

La valeur de w sera donc obtenue au moyen des puissances de 2 et des multiplications. La volonté de choisir la racine unité α comme étant égale à une puissance de 2 est clairement justifiable pour permettre le remplacement de toutes les multiplications par des décalages de bit.

4.5.2 L'implantation Butterfly

De la même façon qu'il existe un algorithme rapide, celui de la transformée de Fourier rapide, pour le calcul de la transformée de Fourier discrète, il existe une version rapide de la transformée en nombres de Fermat. En effet, la même structure VLSI de la *TFR*, de type Butterfly, peut être adoptée pour le calcul d'une *FNT*.

Une FNT de dimension $M = 2^m$, où m est un entier, peut être calculée en utilisant le principe de l'algorithme TFR basé sur la décomposition de Cooley [Cooley1965] :

$$\begin{aligned} X_k &= \left\langle \sum_{n=0}^{M-1} x_n \alpha^{nk} \right\rangle_{F_t} = \left\langle \sum_{n=0}^{\frac{M}{2}-1} x_{2n}\alpha^{2nk} + \sum_{n=0}^{\frac{M}{2}-1} x_{2n+1}\alpha^{(2n+1)k} \right\rangle_{F_t} \\ &= \left\langle \sum_{n=0}^{\frac{M}{2}-1} x_{2n}\alpha^{2nk} + \alpha^k \sum_{n=0}^{\frac{M}{2}-1} x_{2n+1}\alpha^{2nk} \right\rangle_{F_t} \\ &= \left\langle G_k + \alpha^k H_k \right\rangle_{F_t} \end{aligned}$$

avec $k = 0, 1, ..., M-1$. Les vecteurs G_k et H_k sont respectivement les transformées en nombres de Fermat des séquences $\{x_{2n}\}_{n=0}^{\frac{M}{2}-1}$ et $\{x_{2n+1}\}_{n=0}^{\frac{M}{2}-1}$ de longueur $\frac{M}{2}$ chacune. Sachant que $\left\langle \alpha^M \equiv 1 \right\rangle_{F_t}$, nous pouvons écrire :

$$\left\langle \left(\alpha^{M/2}\right)^2 = \alpha^M \equiv 1 = (-1)^2 \right\rangle_{F_t}$$

$$\left\langle \alpha^{M/2} \equiv -1 \right\rangle_{F_t}$$

$$\left\langle \alpha^{k+M/2} \equiv -\alpha^k \right\rangle_{F_t}$$

Nous en déduisons que :

$$\begin{cases} X_k = \left\langle G_k + \alpha^k H_k \right\rangle_{F_t} \\ X_{k+\frac{M}{2}} = \left\langle G_k - \alpha^k H_k \right\rangle_{F_t} \end{cases}, \quad k = 0, 1, ..., \frac{M}{2} - 1$$

Le découpage de la séquence en deux parties peut être répété pour décomposer la transformée de longueur $M = 2^m$ en 4, 8, ..., $\frac{M}{2}$ parties. Par conséquent, la FNT d'une séquence de longueur M peut être calculée à l'aide de $log_2(M)$ décompositions, soit $\frac{Mlog_2(M)}{2}$ transformées de longueur 2.

La figure (4.1) en illustre la structure symétrique, appelée *Butterfly* ou en *papillon*. Le nombre total d'opérations requises pour une transformée en nombres de Fermat est ainsi déterminé à partir de cette structure. Les nombres d'additions et de multiplications mises en oeuvre pour une transformée en nombres de Fermat sont respectivement de l'ordre de $Mlog_2M$ et $\frac{M}{2}log_2M$. Ces multiplications

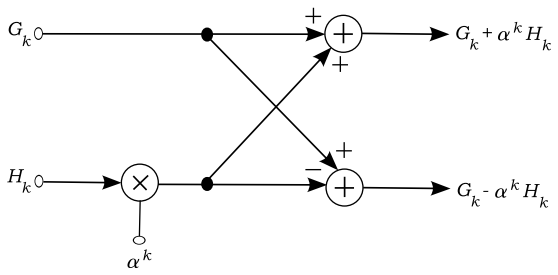

FIGURE 4.1 – Shéma de la décomposition Butterfly

par des puissances de deux sont réalisées par des simples décalages de bits.

4.6 Conclusion

Pour conclure ce chapitre, nous allons résumer les principales caractéristiques de la transformée en nombres de Fermat (*FNT*). La valeur du modulo q égal à un nombre de Fermat permet de choisir des longueurs M égales à des puissances de deux, ce qui favorise une mise en oeuvre de la *FNT* au moyen des mêmes structures Butterfly que celles utilisées par la transformée de Fourier discrète (*TFD*). Pour une implémentation en virgule fixe, la *FNT* présente les avantages suivants par rapport à la *TFD* :

- La possibilité du choix de α égal à une puissance de 2, ce qui permet de remplacer les multiplications par des décalages de bits.
- La simplicité des calculs dans le domaine des entiers dans le cas de l'utilisation de la *FNT* alors que les opérations sur des complexes sont beaucoup plus lourdes à mettre en oeuvre dans le cas de la *TFD*.
- Le calcul de convolution réalisé au moyen de la *FNT* permet d'obtenir un rapport signal à bruit (SNR) plus élevé que celui obtenu par la mise en oeuvre de la *TFD*. En effet, la *TFD* introduit une erreur supplémentaire due à la quantification de ses termes en cosinus et en sinus et à l'arrondissement des calculs intermédiaires alors que les calculs en *FNT* s'effectuent sur des entiers.

Les transformées en nombres de Fermat (*FNT*) admettent la propriété de la convolution cyclique permettant ainsi de les appliquer aux algorithmes de filtrage par blocs overlap-save et overlap-add. La synthèse du filtre bloc optimal dans le cas d'utilisation de la *FNT* fera l'objet de l'étude développée dans le chapitre suivant.

Notons que les $NTTs$, en particulier la FNT, restent un outil de simplification des calculs et n'ont pas de sens physique en elles-mêmes comme celui que possède la TFD dans le domaine fréquentiel.

Chapitre 5

Conception optimale des filtres blocs implémentés en virgule fixe

5.1 Introduction

Le filtrage par blocs, moyennant l'utilisation des transformées rapides pour le calcul des convolutions, a pour objectif de réduire la complexité du filtrage numérique et de réaliser en parallèle le traitement des différents blocs. Dans un calcul de convolution par transformée de Fourier discrète, les calculs intermédiaires, réalisés dans le domaine des complexes $\mathbb{C}$, représente une charge de calcul importante. En effet, une multiplication complexe exige quatre multiplications et deux additions réelles. D'autres transformées, telle que la transformée en cosinus discrète (TCD), peuvent éliminer cet inconvénient puisque leur calcul est réalisé dans le domaine des réels $\mathbb{R}$. Pour une implémentation en virgule flottante (floatting point en anglais), les erreurs de calculs sont négligeables ; les erreurs sur les signaux de sortie sont alors dues uniquement à l'approximation du filtre désiré par un filtre RIF. Pour une implémentation en virgule fixe, deux types d'erreur viennent s'ajouter : l'erreur de quantification des valeurs initiales et l'erreur d'arrondi dans les étapes intermédiaires de calcul [Wong1991]. Bien que les calculs en virgule flottante garantissent un maximum de précision, ils présentent l'inconvénient qu'ils requièrent une charge de calcul très importante et un temps d'exécution des opérations largement supérieur par rapport aux calculs en virgule fixe. Pour les DSP disponibles chez Texas Instrument par exemple, nous pouvons nettement voir cette différence de puissance entre deux modèles de la même génération. Pour deux modèles de la dernière génération conçus pour être cadencés à la même fréquence 600 MHz, le temps pour effectuer une instruction élémentaire est de

3.3 ns pour le modèle TMS320C6727B-300 à virgule flottante alors qu'il est de 1 ns pour le modèle TMS320C6416T à virgule fixe.

Dans ce chapitre, nous proposons une nouvelle approche pour la conception des techniques de filtrage par blocs overlap-save et overlap-add mettant en oeuvre la transformée en nombres de Fermat (FNT). Cette transformée, décrite dans le chapitre précédent, permet une implémentation en virgule fixe de très faible complexité et sans erreurs d'arrondi. Nous présentons ensuite un algorithme très rapide pour la synthèse de ces filtres blocs à moindre distorsion. Cette méthode de synthèse, basée sur la structure matricielle du filtrage par blocs et sur l'utilisation de la propriété de décomposition des matrices circulantes sur le corps de Galois, permet d'obtenir les filtres blocs optimaux selon le critère de minimisation de l'erreur quadratique. Une étude comparative, en terme d'erreurs de distorsion, de coût de filtrage et de coût de synthèse du filtre bloc optimal, du filtrage par blocs utilisant la FNT et celui utilisant la TFD est effectuée. Cette comparaison permet de montrer l'intérêt du filtrage par blocs au moyen de la FNT dans le cas d'une implémentation en virgule fixe.

Notons que la mise en oeuvre des algorithmes en virgule fixe demande la maîtrise de la manière dont le processeur traite les calculs et la vérification de la gamme des valeurs dans les différentes étapes de calcul pour éviter tout effet de débordement de calcul (overflow ou underflow).

5.2 Principe du filtrage OLS et OLA utilisant la FNT

Le principe du filtrage par blocs utilisant la transformée en nombres de Fermat (FNT) est le même que celui du filtrage par blocs utilisant la transformée de Fourier discrète (TFD). Dans ce cas, les convolutions circulaires sont calculées au moyen de la FNT, ce qui permet de réduire le coût de filtrage et de de supprimer les erreurs d'arrondi dans le cas d'une implémentation en virgule fixe.

5.2.1 Overlap-save traité par la FNT

Considérons le cas d'overlap-save dont le principe est illustré sur la figure (5.1). Un bloc de sortie y_k de taille L associé à un bloc d'entrée x_k de taille M est donné par :

$$y_k = S\left\langle T_M^{-1}\left\langle G_s\left\langle T_M x_k\right\rangle_{F_t}\right\rangle_{F_t}\right\rangle_{F_t} \tag{5.1}$$

G_s est la matrice diagonale de taille M telle que $diag\,(G_s) = g_s$, T_M la matrice carrée d'ordre M associée à la FNT $M-$points et T_M^{-1} celle associée à la FNT inverse (IFNT);

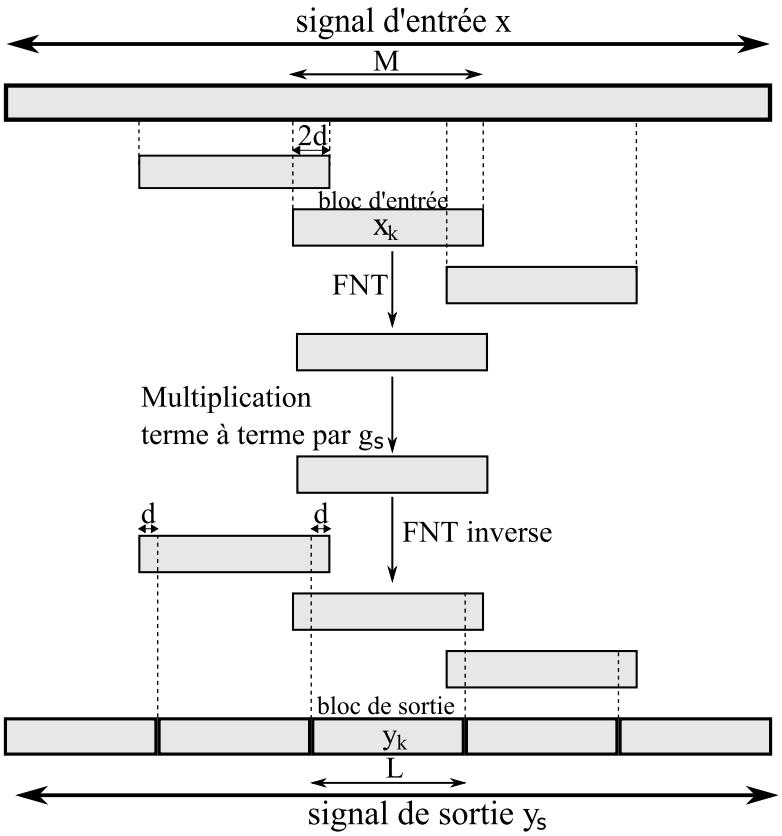

FIGURE 5.1 – Schéma d'overlap-save utilisant la FNT

En utilisant les propriétés de la théorie des nombres rappelées dans le chapitre 4, nous pouvons écrire :

$$y_k = \langle A_s x_k \rangle_{F_t} \tag{5.2}$$

où A_s est la matrice de taille $L \times M$ donnée par :

$$A_s = \left\langle S T_M^{-1} G_s T_M \right\rangle_{Ft} \tag{5.3}$$

Soit $K = bL$, avec b un entier qui désigne le nombre de blocs du signal x. En concaténant les blocs de sortie y_k, $k = 1, 2, ..., b$, le signal global de sortie y_s de taille K, est donné par :

$$y_s = \langle H_s x \rangle_{F_t} \tag{5.4}$$

où H_s est la matrice carrée de taille K formée de b matrices blocs A_s selon la même structure donnée sur la figure 2.2 du chapitre 2.

Le spectre $\overline{y_s}$ du signal de sortie y_s est ainsi donnée par :

$$\overline{y_s} = F_K\left(\langle H_s x\rangle_{F_t}\right) \tag{5.5}$$

5.2.2 Overlap-add traité par la FNT

Pour l'approche overlap-add, le principe est illustré par la figure (5.2).

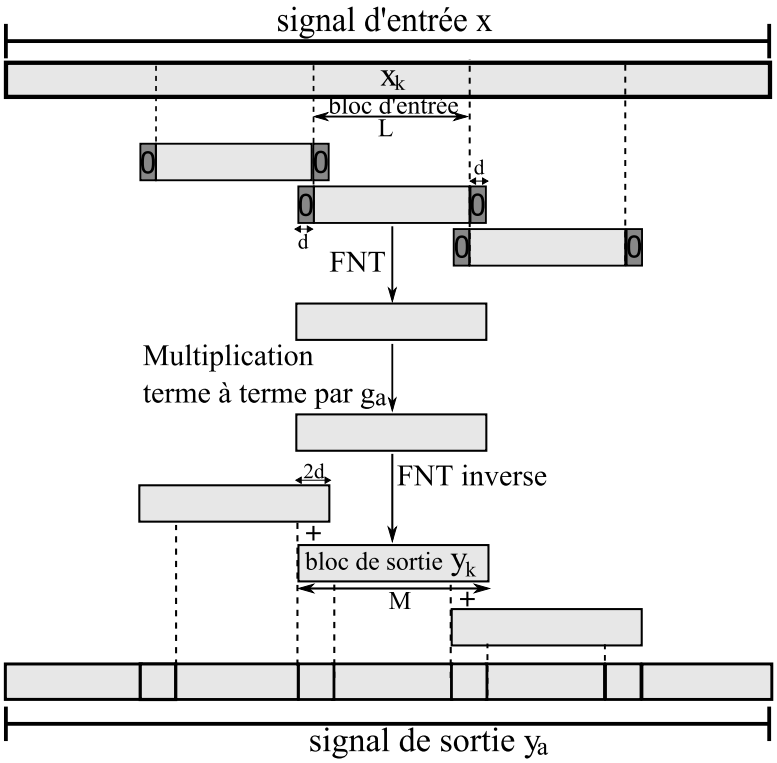

FIGURE 5.2 – Schéma d'overlap-add utilisant la FNT

Un calcul analogue à celui développé dans le cas d'OLS permet d'écrire la relation suivante entre un bloc de sortie y_k de taille M et un bloc d'entrée x_k de taille L :

$$y_k = \langle A_a x_k\rangle_{F_t} \tag{5.6}$$

A_a est la matrice de taille $M \times L$ donnée par :

$$A_a = \left\langle T_M^{-1} G_a T_M S^T\right\rangle_{F_t} \tag{5.7}$$

où G_a est la matrice diagonale de taille M telle que $diag\left(G_a\right) = g_a$. Comme pour le cas d'overlap-save, considérons le cas où $K = bL$, b étant un entier qui désigne le nombre de blocs. Le signal global

de sortie y_a de taille K est donné par :

$$y_a = \langle H_a x \rangle_{F_t} \tag{5.8}$$

où H_a est la matrice carrée de taille K contenant b matrices blocs A_a selon la même structure donnée sur la figure 2.4 du chapitre 2.

Dans ce cas, le spectre $\overline{y_a}$ du signal de sortie y_a est donnée par :

$$\overline{y_a} = F_K \langle H_a x \rangle_{F_t} \tag{5.9}$$

5.2.3 Coût de filtrage

Tel qu'il a été démontré au chapitre précédent, le calcul de la transformée en nombres de Fermat directe ou inverse n'exige aucune opération de multiplication. Ainsi, pour les deux approches OLS et OLA, le traitement de chaque bloc x_k exige :

- Un calcul de FNT directe $M-$points : aucune multiplication.
- M multiplications réelles pour les multiplications terme à terme.
- Un calcul de FNT inverse $M-$points : aucune multiplication.

Ainsi l'obtention de L échantillons de sortie demande M multiplications réelles. Comparée au traitement par TFD demandant un nombre de multiplications réelles de l'ordre de $(2Mlog_2M + 3M)$, l'utilisation de la FNT en filtrage par blocs permet ainsi une réduction importante de complexité.

5.3 Critère d'optimisation

En considérant des tailles M et L des blocs qui sont fixes, l'optimisation du filtrage par blocs overlap-save ou overlap-add porte ainsi sur l'optimisation des coefficients des vecteurs g_s ou g_a. Le critère d'optimisation choisi est celui de la minimisation de l'erreur quadratique moyenne entre le spectre du signal de sortie désiré et le spectre du signal de sortie obtenu par le filtrage par blocs. Ainsi l'erreur à minimiser est donnée par :

$$e_{QM} = \boldsymbol{E}\left\{\sum_{k=0}^{K-1} \left|\overline{y}\,(k) - \overline{y_d}\,(k)\right|^2\right\} = \boldsymbol{E}\left\{\left\|\overline{y} - \overline{y_d}\right\|^2\right\} \tag{5.10}$$

où $\boldsymbol{E}$ désigne l'espérance mathématique, $|.|$ le module d'un complexe et $\|.\|$ la norme euclidienne d'un vecteur. $\overline{y}$ est le spectre du signal de sortie obtenu par filtrage par blocs : $\overline{y_s}$ dans le cas d'overlap-save et $\overline{y_a}$ dans le cas d'overlap-add.

Le principe du filtrage désiré étant le même que celui défini dans le chapitre 2, nous pouvons nous limiter, sans perte de généralité, à des signaux d'entrée x de taille K (voir Chapitre 2, Section 3). Le spectre $\overline{y_d}$ du signal de sortie désiré y_d est donné par :

$$\overline{y_d} = \overline{\overline{H_d}\overline{x}} \tag{5.11}$$

où $\overline{x}$ est le spectre du signal d'entrée x. Le signal de sortie désiré y_d s'écrit alors :

$$y_d = H_d x \tag{5.12}$$

où H_d est la matrice donnée par :

$$H_d = F_K^{-1}\overline{\overline{H_d}}F_K \tag{5.13}$$

Par la suite, nous développons les calculs pour le cas d'overlap-save. L'étude est similaire dans le cas d'overlap-add.

En remplaçant $\overline{y}$ et $\overline{y_d}$ par leurs expressions données respectivement par (5.5) et (5.11), l'équation (5.10) s'écrit :

$$e_{QM} = \boldsymbol{E}\left\{\left\|\left(F_K\left\langle H_s x\right\rangle_{F_t} - \overline{\overline{H_d}\overline{x}}\right)\right\|^2\right\} \tag{5.14}$$

Pour éviter une double interprétation des entiers dans le résultat du calcul de chaque bloc x_k, nous considérons que les blocs de sortie y_k vérifient la condition suivante :

$$y_k = \left\langle A_s x_k\right\rangle_{F_t} = A_s x_k \tag{5.15}$$

En nous basant sur la structure de la matrice H_s, le signal de sortie y_s s'écrit dans ce cas :

$$y_s = \left\langle H_s x\right\rangle_{F_t} = H_s x \tag{5.16}$$

Le spectre $\overline{y_s}$ du signal de sortie y_s s'exprime alors par :

$$\overline{y_s} = F_K y_s = \overline{\overline{H_s}\overline{x}} \tag{5.17}$$

où

$$\overline{\overline{H_s}} = F_K H_s F_K^{-1} \tag{5.18}$$

L'erreur quadratique définie en (5.10) est alors donnée par :

$$e_{QM} = \boldsymbol{E}\left\{\left\|\left(\overline{\overline{H_s}}\overline{x} - \overline{\overline{H_d}}\overline{x}\right)\right\|^2\right\} \tag{5.19}$$

En utilisant l'hypothèse que le signal d'entrée x est un bruit blanc, donc de spectre uniforme, minimiser e_{QM} revient à minimiser l'erreur e donnée par :

$$e = \left\|\overline{\overline{H_s}} - \overline{\overline{H_d}}\right\|_f^2 \tag{5.20}$$

où $\|.\|_f$ désigne la norme de Frobenius d'une matrice.

En remplaçant $\overline{\overline{H}}$ et $\overline{\overline{H_d}}$ par leurs expressions en fonction de H et H_d, et en tenant compte du fait que les matrices F_K et F_K^{-1} associées respectivement à la transformée de Fourier discrète directe et inverse sont unitaires, l'expression de l'erreur e devient :

$$e = \|H_s - H_d\|_f^2 \tag{5.21}$$

Étant donné que la matrice H_d est circulante, et en tenant compte de la structure particulière de la matrice H_s, minimiser e revient à minimiser l'erreur e_1 donnée par :

$$e_1 = \|A_s - A_d\|_f^2 \tag{5.22}$$

A_d est la matrice extraite de la matrice H_d de la même manière que A_s peut être extraite de la matrice H_s.

En remplaçant dans la dernière relation la matrice A_s par son expression donnée par la relation (5.3), nous pouvons écrire :

$$e_1 = \left\|S\left\langle T_M^{-1} G_s T_M \right\rangle_{F_t} - A_d\right\|^2 \tag{5.23}$$

5.4 Problématique

L'expression de l'erreur donnée par la relation (5.23) ressemble à celle obtenue dans les chapitres 2 et 3. Toutefois, la transformée en nombres de Fermat T_M utilisée dans ce chapitre n'est pas une transformation unitaire et la matrice A_s est définie en modulo $A_s = S\left\langle T_M^{-1} G_s T_M \right\rangle_{Ft}$. L'optimisation portant sur les éléments de la diagonale de la matrice G_s doit ramener à des éléments appartenant au corps de Galois $GF(F_t)$. Ce problème d'optimisation ne peut pas être résolu par aucune des approches développées dans les chapitres 2 et 3 car il s'agit d'un problème d'optimisation NP-difficile de type $min_{x\in\mathbb{Z}^n}\left\{\|(Bx-b)\|^2\right\}$, où B est une matrice $m \times n$ avec $m < n$.

Preuve

Sachant que la norme de Frobenius d'une matrice est égale à la norme du vecteur formé par la juxtaposition des lignes de cette matrice, l'erreur e_1 donnée par l'équation (5.22) s'écrit :

$$e_1 = \|ligne\,(A_s) - a_d\|^2 \tag{5.24}$$

où $a_d = ligne(A_d)$ est le vecteur colonne de taille LM formé par la juxtaposition des lignes de la matrice A_d.

En utilisant les propriétés des calculs en modulo et en utilisant les propriétés démontrées dans l'*annexe C*, l'erreur e_1 est donnée par :

$$e_1 = \|\langle F g_s\rangle_{Ft} - a_d\|^2 \tag{5.25}$$

où $g_s = diag\,(G_s)$ est le vecteur inconnu de taille M et F la matrice de taille $LM \times M$ dont les colonnes $F\,(:,i)$ sont données selon la procédure décrite dans l'*annexe C*.

Le problème d'optimisation consiste donc à trouver le vecteur g_s dont les éléments appartiennent au corps de Galois et qui minimise l'erreur e_1 donnée par l'équation (5.25). Ce problème peut être représenté d'une manière générale par :

$$min_x\left\{\left\|\langle Bx\rangle_{F_t} - b\right\|^2\right\} \tag{5.26}$$

où B est une matrice de taille $m \times n$, x et b deux vecteurs de tailles respectives n et m. Les éléments

de la matrice B et du vecteur x sont restreints au domaine de Galois $GF(F_t)$ inclus dans $\mathbb{Z}$. En exprimant le vecteur $\langle Bx \rangle_{F_t}$ sous la forme $\langle Bx \rangle_{F_t} = Bx + F_t r$, où r est un vecteur inconnu de taille m, le système décrit par l'équation (5.26) s'écrit :

$$min_x \left\{ \left\| B' \begin{bmatrix} x \\ r \end{bmatrix} - b \right\|^2 \right\} \tag{5.27}$$

où B' est la matrice de taille $m \times (n+m)$ telle que $B' = \begin{pmatrix} B & \vdots & F_t I_m \end{pmatrix}$. I_m est la matrice identité de taille m.

Ce problème des moindres carrés en nombres entiers est NP-difficile. Le système comporte m équations et $n+m$ inconnues : les n inconnues initiales du vecteur x et les m inconnues du vecteur r. Comme le montre l'étude bibliographique développée ci-après, aucune méthode, autre que la recherche exhaustive, ne permet de résoudre ce problème d'optimisation.

5.5 Problème des moindres carrés en nombres entiers

5.5.1 Définition

Le problème des moindres carrés en nombres entiers (Integer Least Squares ILS en anglais) consiste à trouver le vecteur x dont les éléments appartiennent à l'ensemble des entiers relatifs $\mathbb{Z}$ et qui minimise l'erreur $\|b - Bx\|^2$, où $B \in \mathbb{R}^{n \times m}$ est une matrice réelle de taille $n \times m$ et $b \in \mathbb{R}^m$ est un vecteur réel de taille m :

$$min_{x \in \mathbb{Z}^m} \|b - Bx\|^2 \tag{5.28}$$

Le problème consiste donc à trouver dans l'espace discret $\mathbb{Z}^m$ le vecteur x tel que le vecteur Bx soit le plus proche, au sens des moindres carrés (distance euclidienne), du vecteur b. Comparé au problème des moindres carrés classiques (Least Squares LS en anglais) où $x \in \mathbb{R}^m$ et pour lequel de nombreuses méthodes de résolution sont connues dans la littérature, le problème ILS est un problème NP-difficile [Boas1981, Micciancio2001, Hassibi2005]. Notons que la solution du problème ILS et celle du problème LS classique peuvent être complètement distinctes. Pour illustrer, considérons l'exemple simple où $B = \begin{pmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{pmatrix}$ et $b = (1,1,1)^T$. La solution réelle du problème LS classique est donnée

par $x_{\mathbb{R}} = \left(\frac{-1}{3}, \frac{1}{3}\right)^T$. Cependant, la solution du problème ILS est donnée par $x_{\mathbb{Z}} = (-2, 1)^T$.

Le problème ILS est rencontré dans plusieurs domaines d'applications, en particulier celui des télécommunications, comme les systèmes CDMA ou les systèmes multi-antennes MIMO, celui de la navigation par système de positionnement global (GPS), ou de la cryptographie.

5.5.2 Méthodes de résolution du problème ILS pour le cas des systèmes surdéterminés

Dans le cas où la matrice B, de taille $m \times n$, est de rang n telle que $n \leq m$, le système linéaire est surdéterminé, c.à.d. le nombre d'équations est supérieur au nombre d'inconnues. Plusieurs méthodes sont proposées dans la littérature pourtrouver la solution entière x qui minimise $\|b - Bx\|^2$. Elles sont classées en deux types : exactes et heuristiques.

Méthodes exactes

Le procédé de recherche de la solution optimale par les méthodes exactes se décompose en deux phases. La première, dite phase de réduction, consiste à réduire le système pour obtenir un autre système ILS équivalent mais moins complexe. Des algorithmes de réduction sont développés dans [Lenstra1982] et dans [Lagarias1990]. La deuxième phase, dite de recherche, consiste à rechercher la solution du nouveau système. Deux grandes stratégies de recherche ont été développées dans la littérature. La stratégie dite "Pohst strategy" [Pohst1981, Fincke1985] qui examine le treillis des points situés dans une hyper-sphère et l'autre stratégie dite "Kannan strategy" [Kannan1983, Kannan1987] qui examine le treillis des points situés dans un parallélépipède rectangle. *Schnorr* et *Euchner* ont proposé dans [Schnorr1994] une amélioration de la stratégie "Pohst" fondée sur l'examen des points situés à l'intérieur de l'hyper-sphère dans un ordre différent. Ces différentes stratégies de recherche ont été comparées dans [Agrell2002]. Cette comparaison a montré que la stratégie "Schnorr and Euchner" est la plus rapide. En communication numérique, les méthodes utilisant le procédé de recherche avec les stratégies "Pohst" et "Schnorr-Euchner" pour résoudre le problème ILS sont dites "Sphere Decoder" (SD). Dans la littérature des systèmes GNSS, une autre méthode, appelée LAMBDA (Least-squares AMBiguity Decorrelation Adjustment), est très utilisée pour résoudre le problème ILS. Cette méthode qui a été présentée par *Teunissen* [Teunissen1993, Teunissen1995-a, Teunissen1995-b, Teunissen1999] permet d'obtenir les p premières solutions du problème ILS, p étant un entier positif. La première solution correspond à celle qui donne le résidu minimal, la deuxième

solution est celle qui donne le second résidu minimal et ainsi de suite. Une étude développée dans [Chang2005] a permis de faire la liaison entre la méthode LAMBDA et les méthodes précédemment citées et de proposer une version modifiée de la méthode LAMBDA dite MLAMBDA qui permet de réduire le coût de résolution du problème ILS pour les grandes dimensions.

Méthodes heuristiques

Le problème ILS étant un problème NP-difficile et les méthodes exactes de résolution étant de complexité exponentielle, d'autres méthodes, dites heuristiques ou approximatives, ont été proposées dans la littérature [Wubben2001, Tan2001, Luo2003, Mobasher2005, Mobasher2007]. Les méthodes heuristiques, basées sur la résolution des problèmes des moindres carrés classiques, tiennent leur importance du fait qu'elles permettent d'obtenir une solution sous-optimale avec un coût de calcul largement inférieur à celui exigé par les méthodes exactes. Le principe de ces méthodes heuristiques est décrit ci-dessous.

Un intérêt particulier de ces méthodes réside dans le cas où la matrice B est orthogonale. En effet, dans ce cas, la solution obtenue au moyen de ces méthodes heuristiques est égale à la solution optimale obtenue par les méthodes exactes[Hassibi2005].

Zero-forcing egalization

La méthode heuristique Zero-forcing, de complexité $\mathcal{O}(n^3)$, consiste à résoudre le système dans $\mathbb{R}$ puis à arrondir les éléments réels aux entiers les plus proches. Ainsi, cette méthode heuristique comporte deux phases :

- la première consiste à trouver la solution de l'équation (5.28) dans le corps des réels. Dans ce cas, la solution est donnée au moyen de la méthode de la pseudo-inverse par l'équation :

$$x = \left(B^T B\right)^{-1} B^T b \tag{5.29}$$

- la deuxième phase consiste à arrondir les éléments de x aux entiers les plus proches pour obtenir la solution entière notée x_B et dite estimation de Babai "Babai estimate" [Grotschel1993] :

$$x_B = Round\,(x) \tag{5.30}$$

C'est surtout cette méthode que nous utilisons par la suite dans notre chapitre. Les deux autres méthodes sont décrites à titre indicatif.

Nulling and cancelling

Cette méthode consiste à approximer successivement les coefficients du vecteur x en se basant sur des estimations de Babai. En effet, à partir d'un vecteur x de dimension n, l'estimation de Babai x_B est calculée et une seule parmi les n composantes du vecteur x est alors prise égale à celle de même rang dans le vecteur x_B. Un nouveau système $min_{x' \in \mathbb{Z}^{n-1}} \|B'x' - b\|^2$ peut alors être défini en éliminant du vecteur x cette composante et en supprimant de la matrice B la colonne associée. Le processus d'estimation de Babai est ainsi répété pour estimer les autres composantes, une à chaque itération. En communications numériques, cette méthode correspond à l'égalisation dite "decision-feedback".

Nulling and cancelling with optimal ordering

Une estimation incorrecte de la première composante du vecteur x par la méthode précédente "Nulling and cancelling" peut avoir pour conséquence une mauvaise estimation de toutes les autres composantes du vecteur. Pour minimiser cet effet, une méthode est proposée dans [Foschini1996]. Cette méthode permet d'estimer les composantes du vecteur x dans l'ordre allant de la composante qui a le moins d'effet sur la détermination des autres à celle qui a le plus d'effet. Par exemple, pour une matrice $B = \begin{pmatrix} 1 & 0 & 2 \\ 0 & 2 & 0 \\ 1 & 0 & 3 \\ 1.5 & 0 & 1 \end{pmatrix}$, l'estimation de la deuxième composante du vecteur x n'a aucun effet sur les autres, donc elle est la première composante à être estimée.

5.5.3 Méthodes de résolution du problème ILS pour le cas des systèmes sous-déterminés

Dans certaines applications telles que les systèmes de communications multi antennes, où le canal est considéré linéaire et où le bruit est gaussien, les éléments des messages transmis appartiennent à un certain domaine fini D inclus dans $\mathbb{Z}$. Dans ces applications, le vecteur x, appartenant à un certain sous domaine $D \subset \mathbb{Z}^n$, représente le message transmis, la matrice B de taille $m \times n$

représente le canal de transmission et le vecteur b représente le message bruité reçu. L'algorithme de décodage est réduit à un problème ILS appelé "box-constrained integer least squares (BILS)". Dans le cas où le nombre de lignes m de la matrice B est supérieur au nombre de colonnes n, le problème est dit "box-constrained overdetermined integer least squares (BOILS)", dans l'autre cas où $m \leq n$, le problème est dit "box-constrained underdetermined integer least squares (BUILS)". Bien que plusieurs méthodes de résolution du problème BOILS aient été développées dans la littérature [Damen2003, Boutros2003, Chang2008], peu de méthodes ont été proposées pour la résolution du problème BUILS [Damen2000, Dayal2003, Yang2005, Cui2005]. En outre, ces dernières méthodes de résolution BUILS ne sont applicables que dans le cas où l'on possède des propriétés géométriques liées au vecteur d'inconnues x (module constant des éléments de x) ou des propriétés statistiques liées au canal de transmission (bruit blanc de canal). Dans le cas d'un système BUILS sans aucune propriété, aucune méthode, à part la recherche exhaustive de très grande complexité, permettant d'obtenir la solution optimale du problème, n'est connue dans la littérature.

5.5.4 Cas de notre problème d'optimisation

Dans notre problème, la minimisation de l'erreur e_1 donnée par l'équation (5.23) se réduit à un problème BUILS donnée par l'équation (5.27) sans aucune propriété géométrique ou statistique sur les éléments du vecteur g_s. Ainsi, aucune méthode, à part la recherche exhaustive, ne permet de résoudre ce problème d'optimisation.

Par la suite, nous développons un algorithme analytique de très faible complexité permettant d'obtenir la solution optimale du problème d'optimisation décrit par l'équation (5.23). Cet algorithme est basé sur une propriété importante qui consiste en une décomposition des matrices circulantes sur un corps de Galois $GF(F_t)$ décrite ci-dessous.

5.6 Décomposition des matrices circulantes dans le corps de Galois

5.6.1 Principe de décomposition

Proposition 5.1

Dans le corps de Galois $GF(F_t)$, toute matrice circulante C est diagonalisable et peut se décomposer selon la relation :

$$C = \left\langle T_M^{-1} D T_M \right\rangle_{F_t} \tag{5.31}$$

où D est une matrice diagonale, T_M et T_M^{-1} sont respectivement les matrices associées à la transformée en nombres de Fermat directe (FNT) et inverse (IFNT). La diagonale de la matrice D est obtenue par la FNT directe de la première colonne de la matrice C ou par la FNT inverse de la première ligne selon la relation :

$$diag\,(D) = \langle T_M C(:,0)\rangle_{F_t} = \left\langle M T_M^{-1} C\,(0,:)^T \right\rangle_{F_t} \tag{5.32}$$

Démonstration

Montrons tout d'abord que la matrice C définie par la relation (5.31) est circulante, c'est à dire que $\forall\ (i,j) \in \{0,1,...,M-1\}^2$, nous avons :

$$C\,(i,j) = C\,(\langle i+1\rangle_M, \langle j+1\rangle_M)$$

D'après la relation (5.31), les coefficients de la matrice C sont donnés par :

$$C\,(i,j) = \left\langle \sum_{n=0}^{M-1}\sum_{m=0}^{M-1} T_M^{-1}\,(i,m)\,D\,(m,n)\,T_M\,(n,j) \right\rangle_{F_t}$$

En notant d_n $(n : 0,1,...,M-1)$ les éléments diagonaux de la matrice D, les coefficients $C\,(i,j)$ peuvent s'écrire :

$$C\,(i,j) = \left\langle \sum_{n=0}^{M-1} \left(d_n T_M^{-1}\,(i,n)\,T_M\,(n,j)\right) \right\rangle_{F_t}$$

$$C\,(i,j) = \left\langle \sum_{n=0}^{M-1} \left(d_n M^{-1} \alpha^{\langle -in\rangle_M} \alpha^{\langle nj\rangle_M}\right) \right\rangle_{F_t}$$

$$C\,(i,j) = \left\langle M^{-1} \sum_{n=0}^{M-1} \left(d_n \alpha^{\langle n(j-i)\rangle_M}\right) \right\rangle_{F_t} \tag{5.33}$$

De la même façon, en remplaçant i par $i+1$ et j par $j+1$ dans la dernière équation, nous obtenons :

$$C\,(\langle i+1\rangle_M, \langle j+1\rangle_M) = \left\langle M^{-1} \sum_{n=0}^{M-1} \left(d_n \alpha^{\langle n\left(\langle j+1\rangle_M - \langle i+1\rangle_M\right)\rangle_M}\right) \right\rangle_{F_t}$$

$$C\,(\langle i+1\rangle_M, \langle j+1\rangle_M) = \left\langle M^{-1} \sum_{n=0}^{M-1} \left(d_n \alpha^{\langle n(j-i)\rangle_M}\right) \right\rangle_{F_t}$$

soit, d'après la relation (5.33),

$$C\left(\langle i+1\rangle_M, \langle j+1\rangle_M\right) = C\left(i,j\right)$$

C est donc une matrice circulante.

En remplaçant j par 0 dans l'équation de $C\left(i,j\right)$, les éléments de la première colonne de la matrice C sont donnés par :

$$C\left(i,0\right) = \left\langle M^{-1} \sum_{n=0}^{M-1} \left(d_n \alpha^{\langle -ni\rangle_M}\right) \right\rangle_{F_t} \; ; \; (i:0,1,...,M-1)$$

Sous forme vectorielle, la première colonne de la matrice C est donnée par :

$$C\left(:,0\right) = \left\langle T_M^{-1} diag(D)\right\rangle_{F_t}$$

soit,

$$diag\left(D\right) = \left\langle T_M C(:,0)\right\rangle_{F_t}$$

La proposition énoncée est alors démontrée.

5.7 Algorithme rapide de synthèse du filtre bloc optimal

En tenant compte du fait que la matrice G_s est diagonale, la propriété de décomposition des matrices circulantes sur le corps de Galois permet d'écrire l'erreur e_1 (5.22) sous la forme :

$$e_1 = \|SC_s - A_d\|^2 \tag{5.34}$$

où $C_s = \left\langle T_M^{-1} G_s T_M\right\rangle_{F_t}$ est une matrice circulante dont les éléments appartiennent au corps de Galois $GF(F_t)$. La norme de Frobenius d'une matrice étant égale à la norme euclidienne du vecteur formé la juxtaposition de ses lignes, l'erreur e_1 s'écrit alors :

$$e_1 = \|ligne\left(SC_s\right) - ligne\left(A_d\right)\|^2 \tag{5.35}$$

C_s étant une matrice circulante, toutes ses lignes sont obtenues par un décalage circulaire de sa première ligne, $C_s(i,:)^T = (P_M)^i C_s(0,:)^T$, $i = 0, 1, ..., M-1$. Ainsi, l'erreur e_1 s'écrit :

$$e_1 = \left\| \begin{pmatrix} (P_M)^d \\ (P_M)^{d+1} \\ \vdots \\ (P_M)^{M-d-1} \end{pmatrix} C_s(0,:)^T - ligne(A_d) \right\|^2 \tag{5.36}$$

Le système obtenu est alors un système surdéterminé de taille $LM \times M$ et correspond à un problème de minimisation des moindres carrés en nombres entiers de la forme $min_x \|Bx - b\|^2$ tel que $B = \begin{pmatrix} (P_M)^d \\ (P_M)^{d+1} \\ \vdots \\ (P_M)^{M-d-1} \end{pmatrix}$, $b = ligne(A_d)$ et le vecteur inconnu $x = C_s(0,:)^T$ dont les composantes appartiennent au corps de Galois $GF(F_t) \subset \mathbb{Z}$.

Les colonnes de la matrice B de taille $LM \times M$ étant orthogonales deux à deux (voir Section 5.7.1 ci-après), nous pouvons résoudre ce système ILS par la méthode heuristique Zero-forcing décrite en section 5.5.2. En outre, tel qu'il était démontré en Section 2.9 du Chapitre 2, la structure particulière de la matrice orthogonale B formée par des blocs de matrices permutation permet d'obtenir la solution dans $\mathbb{R}$ au moyen d'un calcul direct ne demandant aucune opération de multiplication . Cette solution est donnée par :

$$C_s(0,:)_{\mathbb{R}}^T = \frac{1}{L} \sum_{i=0}^{L-1} \left(\left(P_M^{-1}\right)^{i+d} A_d(i,:)^T \right) \tag{5.37}$$

P_M^{-1}, inverse de P_M, est la matrice de permutation circulaire associée à l'opération de décalage cyclique à gauche.

La solution entière optimale est ensuite obtenue par l'arrondi de la solution réelle :

$$C_s(0,:)^T = Round\left(C(0,:)_{\mathbb{R}}^T\right) \tag{5.38}$$

Un choix de F_t tel que $F_t > max(A_d(i,j))$ permet d'obtenir $F_t > max\left(C_s(0,:)_{\mathbb{R}}^T\right)$ et par la suite $F_t > max\left(C_s(0,:)^T\right)$. Ainsi les éléments $C_s(0,i)$ appartiennent au corps de Galois $GF(F_t)$.

Pour une implémentation en virgule fixe avec n bits fractionnaires, la solution optimale peut être obtenue par :

$$C_s\left(0,:\right)^T = Round\left(2^n.C\left(0,:\right)_{\mathbb{R}}^T\right) \tag{5.39}$$

Par suite, la diagonale de G_s est obtenue par un calcul de la FNT directe de la première colonne de C_s ou de la FNT inverse sa première ligne :

$$g_s = diag\left(G_s\right) = \left\langle T_M C_s\left(:,0\right)\right\rangle_{F_t} \tag{5.40}$$

$$g_s = diag\left(G_s\right) = \left\langle MT_M^{-1} C_s\left(0,:\right)\right\rangle_{F_t} \tag{5.41}$$

5.7.1 Orthogonalité de la matrice B

Notons E_j, $j : 0 \rightarrow M-1$, les vecteurs colonnes de taille M tels que :

$$\begin{cases} E_j\left(j\right) = 1 \\ E_j\left(i\right) = 0 \qquad i \neq j \end{cases}$$

Les colonnes $P_M\left(:,j\right)$, $j : 0 \rightarrow M-1$, de la matrice P_M peuvent alors s'écrire :

$$P_M\left(:,j\right) = E_{\langle j+1\rangle_M} \quad (j : 0,1,...,M-1)$$

Les colonnes $P_M^m\left(:,j\right)$, $j : 0 \rightarrow M-1$, de la matrice $\left(P_M\right)^m$ s'écrivent alors :

$$\left(P_M\right)^m\left(:,j\right) = E_{\langle j+m\rangle_M} \quad (j : 0,1,...,M-1)$$

Ainsi, les colonnes $B(:,j)$, $j : 0 \rightarrow M-1$, de la matrice $B = \begin{pmatrix} \left(P_M\right)^d \\ \left(P_M\right)^{d+1} \\ \vdots \\ \left(P_M\right)^{M-d-1} \end{pmatrix}$ sont données par :

$$B(:,j) = \begin{pmatrix} E_{\langle j+d\rangle_M} \\ E_{\langle j+d+1\rangle_M} \\ \vdots \\ E_{\langle j+M-d-1\rangle_M} \end{pmatrix} \quad (j : 0,1,...,M-1)$$

Le produit scalaire de deux colonnes j_1 et j_2 de la matrice B est donné par :

$$B(:,j_1).B(:,j_2) = \sum_i B(i,j_1)B(i,j_2) = \begin{pmatrix} E_{\langle d+j_1\rangle_M} \\ E_{\langle d+1+j_1\rangle_M} \\ \vdots \\ E_{\langle M-d-1+j_1\rangle_M} \end{pmatrix} . \begin{pmatrix} E_{\langle d+j_2\rangle_M} \\ E_{\langle d+1+j_2\rangle_M} \\ \vdots \\ E_{\langle M-d-1+j_2\rangle_M} \end{pmatrix}$$

$$B(:,j_1).B(:,j_2) = \sum_{l=d}^{M-d-1} E_{\langle l+j_1\rangle_M}.E_{\langle l+j_2\rangle_M}$$

Sachant que le produit scalaire de deux vecteurs E_{j_1} et E_{j_2} est donné par :

$$E_{j_1}.E_{j_2} = \begin{cases} 1 & pour\, j_1 = j_2 \\ 0 & pour\, j_1 \neq j_2 \end{cases}$$

nous pouvons écrire :

$$B(:,j_1).B(:,j_2) = \begin{cases} M-2d & pour\, j_1 = j_2 \\ 0 & pour\, j_1 \neq j_2 \end{cases}$$

En conclusion, les colonnes de la matrices B sont orthogonales deux à deux.

5.7.2 Coût de synthèse du filtre optimal

Une multiplication ou une division reste l'opération la plus complexe à être réalisée par comparaison à une addition, une soustraction ou un décalage de bits. Ainsi, nous définissons le coût des algorithmes comme étant le nombre de multiplications réelles nécessaires à leur réalisation. Pour un filtrage par blocs utilisant la FNT, notre méthode de synthèse du filtre bloc optimal nécessite de calculer :

- la matrice H_d à partir de laquelle est extraite la matrice A_d : cela demande un calcul TFD $K-$points, donc un nombre de multiplications réelles de l'ordre de $Klog_2K$.
- la colonne $C_s\,(0,:)^T$ de la matrice C_s : cela n'exige aucune multiplication dans la mesure où la solution réelle $C_s\,(0,:)^T_{\mathbb{R}}$ n'exige aucune multiplication et que la solution entière se déduit de celle-ci par simple troncature.
- les coefficients de g_s : cela exige un calcul FNT $M-$points, donc aucune multiplication.

Au total, la complexité de calcul de l'algorithme proposé est de l'ordre de $Klog_2K$ multiplications réelles. Rappelons que dans le cas où la TFD est utilisée, la complexité est de l'ordre de $Klog_2K + Mlog_2M$ multiplications réelles.

5.8 Représentation des résultats

Pour mieux comprendre l'intérêt de l'utilisation de la FNT dans le filtrage par blocs et aussi l'intérêt de notre méthode de conception du filtre bloc optimal, nous présentons et interprétons les résultats obtenus à partir des mêmes hypothèses que celles définies dans les chapitres 2 et 3 : les tailles des blocs sont fixées à $M = 32$ et $L = 24$, la résolution fréquentielle correspond à $K = 96$. La réponse en fréquence du filtre désiré est donnée sur la figure 2.5 du chapitre 2.

5.8.1 Distorsion

Pour une implémentation en virgule fixe, les erreurs de quantification et d'arrondi dépendent du nombre n de bits alloués à la partie fractionnaire des nombres. Nos simulations ont été faites dans le cas d'une représentation en virgule fixe sur 32 bits avec un nombre n de bits de la partie fractionnaire allant de 5 à 10. Notons qu'une augmentation du nombre de bits fractionnaires n, donc de la précision de calcul, permet de réduire la distorsion du filtre mais ceci entraîne une réduction de la gamme dynamique des signaux qui peuvent être traités. Une vérification de la gamme dynamique des signaux traités doit alors être effectuée pour éviter le problème de débordement (overflow). La figure 5.3 représente la distorsion totale des filtres blocs en fonction de la précision et selon la transformée utilisée, TFD ou FNT. Il est clair que lorsque n augmente, la distorsion diminue et converge vers une valeur seuil minimale qui peut être approximée par la valeur de la distorsion obtenue dans le cas d'une représentation en virgule flottante sur 64 bits. Dans le cas de la FNT, cette diminution est liée à l'étape d'arrondi des coefficients du filtre (5.39). Par la suite, nous comparons l'erreur de distorsion LIT et l'erreur de repliement de spectre (aliasing). Le tableau (5.1) résume les différentes erreurs obtenues dans les cas de $n = 5$ et de $n = 10$.

Comparé au filtrage par blocs traité par la TFD, nous constatons que le filtrage par blocs au moyen de la FNT permet de réduire l'erreur de distorsion. Cela est dû au fait que la TFD introduit des erreurs d'arrondi supplémentaires liées à la quantification de ses termes en cosinus et en sinus alors que la FNT, dont les coefficients sont des entiers, permet un calcul sans erreur d'arrondi.

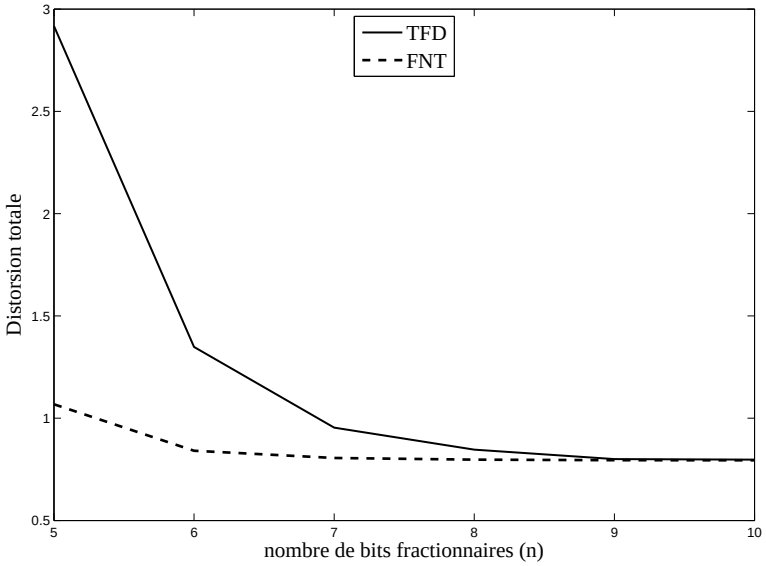

FIGURE 5.3 – Comparaison de la distorsion totale

TABLE 5.1 – Comparaison de la distorsion obtenue selon la transformée utilisée

Implémentation	Précision	Transformée	Distorsion LIT	Aliasing	Distorsion totale
32-bits virgule fixe	$n = 5$	FNT	0.560	0.5	1.068
	$n = 5$	TFD	0.582	1.335	2.917
	$n = 10$	FNT	0.462	0.333	0.795
	$n = 10$	TFD	0.462	0.335	0.797
64-bits virgule flottante		DFT	0.465	0.329	0.794

Notons que, dans la simulation du filtrage par blocs utilisant la TFD, les filtres blocs utilisés sont ceux obtenus par la méthode rapide développée dans le chapitre 2. Ces filtres peuvent être optimisés en rajoutant les contraintes liées à l'implantation en virgule fixe au problème d'optimisation de la synthèse du filtre. Cependant, l'algorithme de synthèse devient dans ce cas un problème de moindres carrés (Integer Least squares) NP-difficile [Kodek1980, Kaufman1977, Salkin1975] puisque la TFD representée en virgule fixe ne correspond pas à une transformation unitaire. Ainsi la propriété de la conservation de la norme euclidienne ou la norme de Frobenius et celle de la diagonalisation des matrices circulantes ne peuvent plus être utilisées dans ce cas.

5.8.2 Coût de synthèse du filtre

Étant donné que la FNT ne demande aucune opération de multiplication, le coût de synthèse d'un filtre bloc FNT est inférieur à celui de synthèse d'un filtre bloc TFD. A partir d'un filtre désiré donné par sa réponse en fréquence K-points, le tableau 5.2 résume le nombre de multiplications

réelles nécessaires à la synthèse du filtre bloc optimal dans le cas de l'utilisation de la TFD et de la FNT ($M = 32$, $L = 24$, $K = 96$).

TABLE 5.2 – Comparaison des coûts de synthèse du filtre optimal selon la transformée utilisée

	Nombres de multiplications réelles requises
FNT	$Klog_2K = 6.32 \times 10^2$
TFD	$Klog_2K + Mlog_2M = 7.92 \times 10^2$

5.8.3 Coût de filtrage

Une réduction très importante du coût de filtrage est obtenue en utilisant la FNT à la place de la TFD. Le tableau 5.3 résume le nombre de multiplications réelles par échantillon de sortie. Une réduction de coût de filtrage de l'ordre de $(2log_2M + 3)$ est obtenue par l'utilisation de la FNT .

TABLE 5.3 – Comparaison des coûts de filtrage selon la transformée utilisée

	Nombres de multiplications réelles requises (par échantillon)
FNT	$M/L = 1.33$
TFD	$(2Mlog_2M + 3M)/L = 17.33$
Réduction	$2log_2M + 3 = 13$

5.9 Conclusion

Le filtrage par blocs utilisant la transformée de Fourier discrète (TFD) est un outil performant pour réduire la complexité de filtrage en communications numériques. Cependant, pour des implémentations en virgule fixe, la transformée de Fourier a pour inconvénient d'introduire sur le filtrage des erreurs supplémentaires dues à la quantification des coefficients de la transformée et aux erreurs d'arrondi. Remplacer la TFD par la transformée en nombres de Fermat (FNT) permet d'un côté de réduire le coût de filtrage et de l'autre côté d'annuler ces erreurs. Dans ce chapitre, nous avons développé un algorithme très rapide pour la synthèse du filtre bloc optimal permettant une implantation en virgule fixe utilisant la FNT. Le critère défini pour l'optimisation est celui de la minimisation de l'erreur quadratique. Cet algorithme rapide est obtenu à partir de la structure du filtrage par blocs et de la décomposition des matrices circulantes dans le corps de Galois que nous avons développée. Nous avons démontré que, pour une implémentation en virgule fixe, l'utilisation de la FNT permet de réduire non seulement la complexité du filtrage et les erreurs de distorsion mais aussi la complexité de synthèse du filtre.

Conclusion et perspectives

Le filtrage à réponse impulsionnelle finie (RIF) occupe une place fondamentale dans le domaine de traitement du signal numérique. Dans les applications nécessitant l'utilisation des filtres de grande longueur, ce filtrage entraîne une lourde charge de calcul, d'où l'intérêt porté aux algorithmes rapides de filtrage. L'overlap-save (OLS) et l'overlap-add (OLA) sont les méthodes les plus efficaces permettant de réduire la complexité arithmétique de ce type de filtrage.

Le travail décrit dans ce rapport de thèse fournit des algorithmes de faible complexité permettant une conception optimale des filtres associés aux techniques de filtrage rapide overlap-save et overlap-add. Ces algorithmes sont développés selon le type d'implémentation envisagée, en virgule flottante ou en virgule fixe. Le critère d'optimisation choisi est celui de la minimisation de l'erreur quadratique de distorsion du filtrage. Les trois principaux axes de notre étude sont : la réduction de la complexité du filtrage, la minimisation des erreurs de distorsion et le développement des algorithmes de faible complexité pour la synthèse des filtres optimaux.

Synthèse du travail de thèse effectué

L'overlap-save et l'overlap-add sont deux techniques de filtrage rapide permettant de réduire considérablement le coût de filtrage. Dans ces techniques, le calcul de la convolution linéaire est remplacé par un calcul de convolution cyclique permettant l'utilisation des transformées rapides telle que la Transformée de Fourier Rapide (TFR). Les notions de base du filtrage RIF, les méthodes classiques de synthèse des filtres RIF ainsi que le principe et l'intérêt des méthodes de filtrage par blocs OLS et OLA ont été rappelés dans le premier chapitre.

Notre travail a consisté dans un premier temps à utiliser les propriétés de l'algèbre linéaire pour

optimiser ces deux techniques de filtrage. Lorsque les tailles des blocs d'entrée et de sortie sont fixes, l'optimisation porte sur les coefficients d'une matrice diagonale qui représente le filtre bloc. Le modèle du filtrage optimal proposé est basé sur un développement de la structure matricielle du filtrage OLS et OLA. Des algorithmes de faible complexité pour la synthèse des filtres optimaux ont été obtenus par l'exploitation des propriétés des matrices circulantes.

Dans le troisième chapitre, nous avons proposé un nouveau modèle de filtrage par blocs dans lequel la matrice qui représente le filtre bloc n'est pas forcément diagonale. Pour ce nouveau modèle, quatre méthodes de synthèse du filtre optimal ont été développées. L'utilisation de l'une de ces méthodes dépend du type de la transformée utilisée dans les calculs de convolution circulaire (unitaire, non unitaire) et de la pondération des fréquences. La nouvelle conception, dite généralisée, permet de réduire davantage les erreurs de distorsion au détriment d'une augmentation du coût de filtrage. Ainsi un compromis entre la distorsion et le coût de filtrage doit être réalisé selon les contraintes liées aux applications envisagées.

Nous avons ensuite évalué la mise en oeuvre de ces algorithmes de filtrage sur des processeurs à virgule fixe. Le choix d'un processeur à virgule fixe est toujours favorisé du fait que son coût de calcul et sa consommation sont largement réduits par rapport à un processeur à virgule flottante. Cependant, les erreurs dues d'une part à la quantification des coefficients du filtre et celle des termes de la transformée utilisée et d'autre part aux arrondis des calculs, peuvent être très importantes dans le cas d'une implémentation en virgule fixe. Pour cela, nous avons approfondi dans le quatrième chapitre les bases mathématiques d'un outil qui peut être utilisé dans diverses applications en traitement du signal lorsqu'il s'agit d'une implémentation en virgule fixe. Cet outil qui est celui des transformées en nombres entiers (NTT : Number Theoretic Transform) a été étudié afin d'extraire quelques propriétés utiles pour le développement et l'optimisation des algorithmes rapides de filtrage. Contrairement à la transformée de Fourier discrète, les calculs en NTT sont effectués sans erreur d'arrondi et les calculs du produit de convolution cyclique sont réalisés sans passage par des calculs dans le domaine des nombres complexes. Nous avons introduit plus particulièrement la transformée en nombres de Fermat (FNT : Fermat Number Transform) qui permet un calcul très rapide des produits de convolution circulaire et qui peut être mise en oeuvre sur une structure Butterfly similaire à celle de la transformée de Fourier rapide.

Dans le dernier chapitre, nous avons développé une nouvelle approche de conception optimale du filtrage OLS et OLA mettant en oeuvre la transformée en nombres de Fermat. En nous basant

sur les propriétés de la théorie des nombres et sur le développement matriciel de la structure du filtrage, un algorithme de synthèse de très faible complexité a été développé.

Les résultats de ce travail peuvent être utilisés dans les applications temps-réel afin de concevoir des filtres optimaux et de faible complexité pour l'implantation des algorithmes de filtrage. Ces résultats ont fait l'objet de la publication de deux articles IEEE [Daher2008, Daher2009-a], d'une communication internationale ICASSP-2009 [Daher2009-b] et d'un article en cours de soumission.

Perspectives du travail de thèse

Dans toute notre étude, l'optimisation du filtrage était basée sur le critère de la minimisation de l'erreur quadratique. Un premier travail qu'il serait intéressant d'aborder est de compléter cette étude par d'autres basées sur d'autres critères d'optimisation, par exemple le critère de Chebyshev.

Dans le but d'obtenir une réduction des erreurs de distorsion sans pour autant augmenter la complexité de filtrage et le coût de synthèse des filtres optimaux, une étude plus détaillée de la conception généralisée du filtrage par blocs mériterait d'être développée. Une conception d'un filtrage optimal dans lequel une partie ou l'ensemble des éléments de la matrice représentant le filtre bloc sont des puissances de deux par exemple pourrait être utile dans cet objectif.

Une autre perspective au niveau pratique pourrait être le développement des applications mettant en oeuvre les algorithmes développés dans ce mémoire. En particulier, l'implantation du filtrage par blocs mettant en oeuvre la FNT sur une carte FPGA ou des DSPs à virgule fixe basés sur les circuits en chaîne "Pipeline" de la structure Butterfly pourra conduire à une diminution significative de la surface matérielle utilisée tout en permettant de travailler en temps réel avec des durées réduites de calcul. En effet, l'implémentation des fonctions réalisées en virgule fixe modulo un nombre de Fermat est basée sur des opérateurs logiques de bases de type ET,OU, $NON-ET$ et $OU\ Exclusif$.

Annexe A

Démonstration de la propriété 2.5

Propriété 2.5

Le repliement de spectre d'une fréquence de sortie i peut être causé uniquement par les fréquences d'entrée j telles que :

$$j - i = q\frac{K}{L}, \quad q \in \mathbb{Z}^{\star}$$

soit,

$$\forall\,(i,j)\,,\left(j-i \neq q\frac{K}{L},\ q\in\mathbb{Z}^{\star}\right) \Rightarrow \left\{\begin{array}{rcl} \overline{\overline{H_s}}\,(i,j) & = & 0 \\ \overline{\overline{H_a}}\,(i,j) & = & 0 \end{array}\right.$$

Démonstration

Cette propriété est démontrée pour le cas d'OLS. Elle peut être démontrée de la même manière dans le cas d'OLA.

En se référant à l'équation (2.28), et à la structure de H_s en fonction de A_s donnée par la figure (2.2), nous pouvons écrire :

$$\overline{\overline{H_s}}\,(i,j) = \frac{1}{K}\sum_{m=0}^{L-1}\sum_{n=0}^{M-1}\sum_{l=0}^{b-1}\alpha_K^{-im+jn+lL(j-i)-jd}A_s\,(m,n)$$

$$\overline{\overline{H_s}}\,(i,j) = \frac{1}{K}\alpha_K^{-jd}\sum_{m=0}^{L-1}\sum_{n=0}^{M-1}\left(\alpha_K^{-im+jn}\sum_{l=0}^{b-1}u^l A_s\,(m,n)\right)$$

où $u = \alpha_K^{L(j-i)}$. Or $\alpha_K^{L(j-i)}$ est une racine $L^{ième}$ de l'unité, par conséquent :

$$\begin{cases} \sum_{l=0}^{b-1} u^l &= 0 \ \ pour\, u \neq 1 \\ \sum_{l=0}^{b-1} u^l &= b \ \ pour\, u = 1 \end{cases}$$

soit,

$$\overline{\overline{H_s}}(i,j) = \begin{cases} 0 & pour\, u \neq 1 \\ \frac{b}{K} \alpha_K^{-jd} \sum_{m=0}^{L-1} \sum_{n=0}^{M-1} \alpha_K^{-im+jn} A_s(m,n) & pour\, u = 1 \end{cases}$$

Or $u = \alpha_K^{L(j-i)} \neq 1$ pour :

$$L(j-i) \neq qK, \quad q \in \mathbb{Z}$$

soit,

$$j - i \neq q\frac{K}{L}, \quad q \in \mathbb{Z}$$

Ainsi l'effet de repliement de spectre de la composante spectrale $\overline{x}(j)$ du signal d'entrée est nul pour les composantes spectrales $\overline{y}(i)$ du signal de sortie telles que $j - i \neq a\frac{K}{L}$.

Annexe B

Démonstration de la propriété 2.6

Propriété 2.6

En OLS, les fréquences d'entrée j telles que $j = t\frac{K}{pgcd(K,M)}$, $t \in \mathbb{Z}$ n'ont aucun effet de repliement de spectre, soit :

$$\forall\, i \neq j,\ \left(j = t\frac{K}{pgcd\,(K,M)},\ t \in \mathbb{Z}\right) \Rightarrow \overline{\overline{H_s}}\,(i,j) = 0$$

En OLA, les fréquences de sortie i telles que $i = t\frac{K}{pgcd(K,M)}$, $t \in \mathbb{Z}$ ne subissent aucun effet de repliement de spectre, soit :

$$\forall\, i \neq j,\ \left(i = t\frac{K}{pgcd\,(K,M)},\ t \in \mathbb{Z}\right) \Rightarrow \overline{\overline{H_a}}\,(i,j) = 0$$

Démonstration

Cette propriété est démontrée pour le cas d'OLA. Elle peut être démontrée de la même manière dans le cas d'OLS.

En se référant à l'équation (2.35), et à la structure de H_a en fonction de A_a donnée par la figure (2.4), nous pouvons écrire :

$$\overline{\overline{H_a}}\,(i,j) = \frac{1}{K}\alpha_K^{id}\sum_{m=0}^{M-1}\sum_{n=0}^{L-1}\left(\alpha_K^{-im+jn}\sum_{l=0}^{b-1}u^l A_a\,(m,n)\right)$$

Sachant que $\overline{\overline{H_a}}\,(i,j) = 0$ pour $j - i \neq q\frac{K}{L}$, $q \in \mathbb{Z}$ (Propriété 2.5, Annexe A), les autres termes pour

lesquels $j-i=q\frac{K}{L}$, $q\in\mathbb{Z}$ sont donnés par :

$$\overline{\overline{H_a}}(i,j)=\frac{b}{K}\alpha_K^{id}\sum_{m=0}^{L-1}\sum_{n=0}^{M-1}\alpha_K^{-im+jn}A_a(m,n)$$

soit,

$$\overline{\overline{H_a}}\left(i,i+q\frac{K}{L}\right)=\gamma\sum_{n=0}^{L-1}\alpha_K^{q\frac{K}{L}n}\sum_{m=0}^{M-1}\alpha_K^{i(n-m)}A_a(m,n)$$

où $\gamma=\frac{b}{K}\alpha_K^{id}$.

Sachant que $A_a(m,n)=C_a(m,n+d)=c_{\langle n-m+d\rangle_M}$, nous obtenons :

$$\overline{\overline{H_a}}\left(i,i+q\frac{K}{L},\right)=\gamma\sum_{n=0}^{L-1}\alpha_K^{q\frac{K}{L}n}\sum_{p=n-M+1}^{n}\alpha_K^{ip}c_{\langle p+d\rangle_M}$$

Posons

$$\beta(n)=\sum_{p=n-M+1}^{n}\alpha_K^{ip}c_{\langle p+d\rangle_M}$$

soit,

$$\beta(n)=\sum_{p=n-M+1}^{-1}\alpha_K^{ip}c_{\langle p+d\rangle_M}+\sum_{p=0}^{n}\alpha_K^{ip}c_{\langle p+d\rangle_M}$$

$$\beta(n)=\sum_{p'=n+1}^{M-1}\alpha_K^{i(p'-M)}c_{\langle p'+d\rangle_M}+\sum_{p=0}^{n}\alpha_K^{ip}c_{\langle p+d\rangle_M}$$

Dans le cas où $\alpha_K^{-iM}=1$, $\beta(n)$ s'écrit :

$$\beta(n)=\sum_{p=0}^{M-1}\alpha_K^{ip}c_{\langle p+d\rangle_M}$$

Par conséquent, nous obtenons :

$$\overline{\overline{H_a}}\left(i,i+q\frac{K}{L},\right)=\gamma\sum_{n=0}^{L-1}\alpha_K^{q\frac{K}{L}n}\sum_{p=0}^{M-1}\alpha_K^{ip}c_{\langle p+d\rangle_M}$$

soit,

$$\overline{\overline{H_a}}\left(i,i+q\frac{K}{L},\right)=\gamma\sum_{p=0}^{M-1}\alpha_K^{ip}c_{\langle p+d\rangle_M}\sum_{n=0}^{L-1}\alpha_K^{q\frac{K}{L}n}$$

Or $\alpha_K^{\frac{qKn}{L}}$ est une racine $L^{ième}$ de l'unité, par conséquent :

$$\sum_{n=0}^{L-1} \alpha_K^{\frac{qKn}{L}} = \sum_{n=0}^{L-1} \left(\alpha_K^{\frac{qK}{L}} \right)^n = 0$$

Ainsi,

$$\overline{\overline{H_a}} \left(i, i + q\frac{K}{L}, \right) = 0 \;\; si \; \alpha_K^{-iM} = 1$$

Ceci est le cas pour

$$iM = pK, \;\; p \in \mathbb{Z}$$

Soient λ, μ et κ les trois entiers tels que : $\lambda = pgcd\,(M, K)$, $M = \mu\lambda$ et $K = \kappa\lambda$, la condition précédente se traduit par :

$$i\mu = p\kappa$$

μ et κ étant premiers entre eux, il en déduit, d'après le théorème de Gauss, que κ divise i, soit,

$$i = t\kappa, \; t \in \mathbb{Z}$$

soit,

$$i = t\frac{K}{\lambda} = t\frac{K}{pgcd\,(M, K)}$$

Annexe C

Détermination des matrices F et E des méthodes 1 et 3 du chapitre 3

En notant Δ_{mn} les matrices carrés de taille M telles que :

$$\Delta_{mn}(k,l) = \delta_{(m,k),(n,l)} = \begin{cases} 1 \quad si \quad m = k \ et \ n = l \\ 0 \qquad\qquad sinon \end{cases} \tag{C.1}$$

Les matrices G, A, H et $\overline{\overline{H}}$ s'écrivent :

$$G = \sum_{m=0}^{M-1} \sum_{n=0}^{M-1} G(m,n)\Delta_{mn} \tag{C.2}$$

$$A = ST_M^{-1} G T_M = \sum_{m=0}^{M-1} \sum_{n=0}^{M-1} G(m,n) A_{mn} \tag{C.3}$$

$$H = \sum_{m=0}^{M-1} \sum_{n=0}^{M-1} G(m,n) H_{mn} \tag{C.4}$$

$$\overline{\overline{H}} = T_K H T_K^{-1} = \sum_{m=0}^{M-1} \sum_{n=0}^{M-1} G(m,n) \overline{\overline{H_{mn}}} \tag{C.5}$$

où

- A_{mn} est la matrice obtenue à partir de Δ_{mn}, soit $A_{mn} = ST_M^{-1}\Delta_{mn}T_M$.
- H_{mn} est la matrice construite à partir de la matrice A_{mn} selon la structure donnée par la

figure 2.2 du chapitre 2.

– $\overline{\overline{H_{mn}}}$ est la matrice obtenue à partir de H_{mn}, soit $\overline{\overline{H_{mn}}} = T_K H_{mn} T_K^{-1}$.

Il en résulte :

$$ligne\,(A) = \sum_{m=0}^{M-1}\sum_{n=0}^{M-1} G(m,n) ligne\,(A_{mn}) \tag{C.6}$$

$$ligne\,(H) = \sum_{m=0}^{M-1}\sum_{n=0}^{M-1} G(m,n) ligne\,(H_{mn}) \tag{C.7}$$

$$ligne\left(\overline{\overline{H}}\right) = \sum_{m=0}^{M-1}\sum_{n=0}^{M-1} G(m,n) ligne\left(\overline{\overline{H_{mn}}}\right) \tag{C.8}$$

$$ligne\left(Z\overline{\overline{H}}\right) = \sum_{m=0}^{M-1}\sum_{n=0}^{M-1} G(m,n) ligne\left(Z\overline{\overline{H_{mn}}}\right) \tag{C.9}$$

où $ligne(A)$ est le vecteur formé par la juxtaposition des lignes de la matrice A.

Les équations (C.6) et (C.9) peuvent s'écrire :

$$ligne\,(A) = Eg \tag{C.10}$$

$$ligne\left(Z\overline{\overline{H}}\right) = Fg \tag{C.11}$$

où g est le vecteur colonne formé par la juxtaposition des éléments $G(m,n)$ de la matrice G. E est la matrice de taille $LM \times M^2$ dont les colonnes sont les vecteurs données par $ligne\,(A_{mn})$. F est la matrice de taille $K^2 \times M^2$ dont les colonnes sont les vecteurs données par $ligne\left(Z\overline{\overline{H_{mn}}}\right)$.

Dans le cas où les coefficients de G ne sont pas tous libres, les matrices E et F peuvent être réduits en éliminant les colonnes correspondantes aux éléments nuls du vecteur g. Par exemple, dans le cas d'une matrice G diagonale, le vecteur g est donné par $g = diag\,(G)$ et E F sont des matrices à M colonnes telle que $E\,(:,m) = ligne\,(A_{mm})$ et $F\,(:,m) = ligne\left(Z\overline{\overline{H_{mm}}}\right)$.

Coût de calcul de F :

Chaque colonne de la matrice F exige les calculs suivants :

– Calcul de la matrice A_{mn} : le nombre de multiplications nécessaires dépend de la transformée T_M utilisée. Sachant que la valeur de M est souvent très inférieure à celle de K, nous pouvons négliger le coût de ce calcul.

- Construction de la matrice H_{mn} : ceci ne nécessite aucune opération de multiplication.
- Calcul de $\overline{\overline{H_{mn}}} = F_K H_{mn} F_K^{-1}$: ceci nécessite une TFD à deux dimensions, soit K TFDs sur les colonnes de H_{mn} suivies de K TFDs sur les lignes de la matrice obtenue ($F_K H_{mn}$), soit un nombre de multiplications réelles de l'ordre $2K^2 log_2 K$.
- Calcul de la matrice $Z\overline{\overline{H_{mn}}}$: sachant que Z est une matrice diagonale à éléments réelles, ceci nécessite $2K^2$ multiplications réelles.

Au total, le coût de calcul de chaque colonne de F est de l'ordre de $2K^2\,(log_2 K + 1)$ multiplications réelles.

Annexe D

Détermination de la matrice Ω_β de la méthode 2 du chapitre 3

Selon les structures des matrices Δ_{mn} définies en *Annexe C*, nous pouvons écrire :

$$\Delta_{mn}(l,k) = \Delta_{00}\left(\langle l-m\rangle_M, \langle k-n\rangle_M\right), \quad (l,k=0,1,...,M-1) \tag{D.1}$$

Basé sur la propriété du retard de la TFD, les matrices $C_{mn} = F^{-1}\Delta_{mn}F$ et $C_{00} = F^{-1}\Delta_{00}F$ sont liées par :

$$C_{mn}(l,k) = e^{+j\frac{2\pi(lm-kn)}{M}} C_{00}(l,k) \quad (l,k=0,1,...,M-1) \tag{D.2}$$

Δ_{00} étant une matrice diagonale, l'obtention de la matrice C_{00} et parsuite A_{00}, H_{00}, et $H_{00}^{\star}$, ne nécessite qu'une une TFD de taille M.

Les matrices H_{mn} et H_{00} construites respectivement à partir des matrices $A_{mn} = SC_{mn}$ et $A_{00} = SC_{00}$ selon la structure 2.2 du chapitre 2 sont alors liées par :

$$H_{mn}(l,k) = e^{+j\frac{2\pi(lm-kn)}{M}} H_{00}(l,k) \quad (l,k=0,1,...,K-1)$$

soit

$$H_{mn}(l,k) = e^{+j\frac{2\pi\left(l\frac{Km}{M}-k\frac{Kn}{M}\right)}{K}} H_{00}(l,k) \quad (l,k=0,1,...,K-1) \tag{D.3}$$

Supposons que M divise K. La propriété de retard de la TFD permet alors d'établir entre $\overline{\overline{H_{mn}}} =$

$F_K H_{mn} F_K^{-1}$ et $\overline{\overline{H_{00}}} = F_K H_{00} F_K^{-1}$ la relation suivante :

$$\overline{\overline{H_{mn}}}(l,k) = \overline{\overline{H_{00}}}\left(\langle l - l_m \rangle_K, \langle k - k_n \rangle_K\right), \quad (l,k = 0,1,...,K-1) \tag{D.4}$$

avec $l_m = \frac{Km}{M}$ et $k_n = \frac{Kn}{M}$.

Sachant que les colonnes $F_\alpha, \alpha = 0,1,...,s-1$, de la matrice F s'obtiennent par $F_\alpha = ligne\left(Z\overline{\overline{H_{m_\alpha n_\alpha}}}\right)$, les éléments $(F^\star F)(\alpha,\beta) = F_\alpha^\star F_\beta$, $\alpha,\beta = 0,1,...,s-1$, de la matrice $F^\star F$ sont alors données par :

$$(F^\star F)(\alpha,\beta) \quad = \quad ligne\left(\overline{\overline{H_{m_\alpha n_\alpha}}}^\star Z^\star\right) \bullet ligne\left(Z\overline{\overline{H_{m_\beta n_\beta}}}\right)$$

Z étant une matrice diagonale, nous pouvons écrire :

$$(F^\star F)(\alpha,\beta) = ligne\left(\overline{\overline{H_{m_\alpha n_\alpha}}}^\star\right) \bullet ligne\left(Z\overline{\overline{H_{m_\beta n_\beta}}}Z^\star\right)$$

soit,

$$(F^\star F)(\alpha,\beta) = \sum_l \sum_k \overline{\overline{H_{m_\alpha n_\alpha}}}^\star (l,k)\, \overline{\overline{H_\beta}}(l,k)$$

où $\overline{\overline{H_\beta}} = Z\overline{\overline{H_{m_\beta n_\beta}}}Z^\star$.

D'après l'équation (D.4), cette dernière relation peut s'écrire :

$$(F^\star F)(\alpha,\beta) = \sum_l \sum_k \overline{\overline{H_{00}}}^\star \left(\langle l - l_\alpha \rangle_K, \langle k - k_\alpha \rangle_K\right) \overline{\overline{H_\beta}}(l,k)$$

où $l_\alpha = \frac{Km_\alpha}{M}$ et $k_\alpha = \frac{Kn_\alpha}{M}$.

En notant Ω_β la matrice carrée d'ordre K dont les éléments sont donnés par la convolution circulaire de dimension deux :

$$\Omega_\beta(l_\alpha, k_\alpha) = \sum_l \sum_k \overline{\overline{H_{00}}}^\star \left(\langle l - l_\alpha \rangle_K, \langle k - k_\alpha \rangle_K\right) \overline{\overline{H_\beta}}(l,k)$$

elle peut être obtenue par la relation :

$$\Omega_\beta = F_K^{-1}\left(H_{00}^\star \boxdot H_\beta\right) F_K \tag{D.5}$$

Les éléments $F^\star F(\alpha,\beta)$, $\alpha = 0,1,...,s-1$, de la $\beta^{ème}$ colonne de la matrice $F^\star F$ sont donc les

coefficients $\Omega_\beta(l_\alpha, k_\alpha)$ de la matrice Ω_β tels que $l_\alpha = \frac{Km_\alpha}{M}$ et $k_\alpha = \frac{Kn_\alpha}{M}$.

Coût de calcul de Ω_β :

Hormis le calcul de $H_{00}^\star$ et de $\overline{\overline{H_{00}}}$, chaque matrice Ω_β exige les calculs suivants :

- $\overline{\overline{H_{m_\beta n_\beta}}}$ obtenue à partir de $\overline{\overline{H_{00}}}$ (voir *annexe A*) : ce calcul n'exige aucune multiplication.
- $\overline{\overline{H_\beta}} = Z\overline{\overline{H_{m_\beta n_\beta}}}Z^\star$: ce calcul exige un nombre de multiplications réelles de l'ordre de $4K^2$.
- H_β obtenue par une TFD à deux dimensions de $\overline{\overline{H_\beta}}$: ce calcul exige un nombre de multiplications réelles de l'ordre $2K^2 log_2 K$.
- $H_{00}^\star \boxdot H_\beta$: ce calcul exige K^2 multiplications complexes, soit $3K^2$ multiplications réelles.
- $F_K^{-1}(H_{00}^\star \boxdot H_\beta) F_K$ obtenue par une TFD à deux dimensions de $H_{00}^\star \boxdot H_\beta$: ce calcul exige donc un nombre de multiplications réelles de l'ordre $2K^2 log_2 K$.

Le calcul de H_{00} ne nécessite qu'une une TFD de taille M, soit un nombre de multiplications réelles de l'ordre $M log_2 M$. En négligeant le coût de calcul de $H_{00}^\star$ et de $\overline{\overline{H_{00}}}$ (M est souvent très inférieure à K), le calcul de chaque matrice Ω_β exige alors un nombre de multiplications réelles de l'ordre $4K^2 log_2 K + 7K^2$.

Annexe E

Corps de Galois

Par définition, un corps de Galois $GF(q)$ d'ordre égal à un entier q est constitué de q éléments.

- Si q est premier, le corps de Galois GF (q) est constitué de q éléments de l'ensemble $\{0, ..., q-1\}$. $GF(q)$ sera dit primitif.
- Si q n'est pas premier, il existe un nombre p tel que $q = p^n$ où p est premier et n entier. $GF(p^n)$ est alors appelé extension du corps $GF(p)$.

Tous les éléments d'un corps de Galois $GF(q)$ peuvent être additionnés, soustraits, multipliés et divisés ; les opérations s'effectuant modulo q. La somme ou le produit de n'importe quels éléments de $GF(q)$ est aussi un élément du corps de Galois $GF(q)$.

Pour tout élément k de $GF(q)$, tel que $k \neq 0$, il existe un élément opposé noté $-k$ tel que : $k + (-k) = 0$. Ainsi l'opération de soustraction d'un élément l de $GF(q)$ est définie par : $k-l = k + (-l)$ où $(-l)$ est l'opposé de l.

Pour tout élément k de $GF(q)$, tel que $k \neq 0$, il existe un inverse unique k^{-1}, tel que : $k.k^{-1} = 1$. Ainsi l'opération de division, par un élément l de $GF(q)$ est définie par : $\frac{k}{l} = k \times l^{-1}$ où l^{-1} est l'inverse de l.

Dans un corps de Galois (GF (q) , $+$,$\times$) , les propriétés d'associativité, de commutativité et de distributivité sont vérifiées.

Annexe F

Algorithme d'Euclide étendu

L'algorithme d'Euclide étendu est une variante de l'algorithme d'Euclide qui permet, à partir de deux entiers a et b, de calculer non seulement leur plus grand commun diviseur, mais aussi un de leurs couples de coefficients de Bézout qui sont deux entiers u et v tels que :

$$au + bv = PGCD(a, b) \tag{F.1}$$

Etant donné deux entiers a et b, l'algorithme d'Euclide permet de trouver l'unique quotient q et l'unique reste r tels que $a = qb + r$, avec $r < b$. En effet, q et r peuvent être obtenus en procédant à une série de divisions successives :

$$\begin{cases} a = q_1 b + r_2 & r_2 < b \\ b = q_2 r_2 + r_3 & r_3 < r_2 \\ \vdots & \\ r_{p-2} = q_{p-1} r_{p-1} + r_p & r_p < r_{p-1} \\ r_{p-1} = q_p r_p + 0 & \end{cases} \tag{F.2}$$

L'algorithme s'arrête lorsque $r_p = 0$. Le plus commun diviseur de u et v, noté $PGCD\,(a, b)$, est donné par le dernier reste non nul r_p. Cet algorithme peut être présenté sous la forme de divisions successives pour obtenir les restes $r_i = rem\,(r_{i-2}, r_{i-1})$ avec $r_0 = a$ et $r_1 = b$.

Observons que a et b apparaissent sur les deux premières lignes. D'autre part, la valeur la plus à droite dans chaque ligne (à partir de la $2^{ème}$ ligne) est le reste de la ligne précédente, et le dividende, dans chaque égalité à partir de la $3^{ème}$ ligne, est le reste obtenu deux lignes plus haut. Nous pouvons

ainsi calculer progressivement chaque reste successif comme combinaison linéaire des deux valeurs initiales a et b :

$$\begin{cases} r_1 = a - q_1 b \\ r_2 = b - q_2 r_1 = (1 + q_1 q_2)\, b - q_2 a \\ \vdots \\ r_{p-1} = au + bv \end{cases} \tag{F.3}$$

Les entiers u et v peuvent s'obtenir par récursivité par :

$$\begin{cases} Q_i = ent\left(\frac{r_{i-2}}{r_{i-1}}\right) \\ u_i = u_{i-2} - Q_i u_{i-1} \\ v_i = v_{i-2} - Q_i v_{i-1} \end{cases} \tag{F.4}$$

où les données initiales sont données par $u_0 = v_1 = 1$ et $u_1 = v_0 = 0$.

Quand a et b sont premiers entre eux $PGCD(a, b) = r_{p-1} = 1$, u est alors l'inverse pour la multiplication de a modulo b, ce qui est utilisé pour le calcul d'inverse dans le corps de Galois.

Bibliographie

[Agarwal1974] R. C. Agarwal and C.S. Burrus, "Fast Convolution Using Fermat Number Transforms with Applications to Digital Filtering", IEEE Trans. on Acoustics, Speech and Signal Processing, Vol. ASSP-22, No. 2, pp. 87-97, 1974.

[Agarwal1975] R. C. Agarwal and C.S. Burrus, "Number Theoretic Transforms to Implement Fast Digital Convolution", IEEE Proceedings, Vol.63, No. 4, pp. 550-560, April. 1975.

[Agrell2002] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger. Closest point search in lattices. IEEE Transactions on Information Theory, Vol. 48, No. 8, pp. 2201-2214, 2002.

[Alfredsson1996] L-I. Alfredsson, "VLSI Architecture and Arithmetic Operations with Applications to the fermat Number transform", Linkoping University, Sweden 1996.

[Andraka2004] R. Andraka, "Supercharge Your DSP with Ultra-Fast Floating-Point FFTs", DSP magazine, pp. 42-44, April 2007.

[Antoniou1979] A. Antoniou, Digital Filters: Analysis and Design. New York: Mc- Graw-Hill, 1979.

[Blahut1985] R. Blahut, "Fast algorithms for digital signal processing", Addison-Wesley Publishing Company, 1985.

[Bloomfield1976] P. Bloomfield, Fourier Analysis of Time Series: An introduction, John Wiley & Sons, New York, 1976.

[Boas1981] P. van Emde Boas, "Another NP-complete partition problem and the complexity of computing short vectors in a lattice", Technical report 81-04, University of Amsterdam, Mathematisch Institute, The Netherlands, 1981.

[Boutros2003] J. Boutros, N. Gresset, L. Brunel, and M. Fossorier, "Soft-input soft-output lattice sphere decoder for linear channels", In Proceedings of IEEE 2003 Global Communications Conference, pp. 213-217, San Francisco, U.S.A., December 2003.

[Burel2004] G. Burel, "Optimal design of transform-based block digital filters using a quadratic criterion", IEEE Trans. Signal Processing, Vol. 52, No. 7, pp. 1964-1974, Jul. 2004.

[Chang2005] X.-W. Chang, X. Yang, and T. Zhou, "MLAMBDA : a modified LAMBDA method for integer least-squares estimation", Journal of Geodesy, Vol. 79, pp.552-565, 2005.

[Chang2008] X.-W. Chang and Q. Han, "Solving box-constrained integer least-squares problems", IEEE Transactions on Wireless Communications, Vol. 7, No. 1, pp. 277-287, January 2008.

[Chauvet2004] W. Chauvet, "Etude des filtres LPTV numériques,Applications aux communications numériques", Thèse de doctorat, Ecole Doctorale d'Informatique et de Télécommunications, Institut National Polytechnique de Toulouse, November 2004.

[Chevillat1978] P. R. Chevillet, "Transform-Domain Digital Filtering with Number Theoretic Transforms and Limited Word Lenghts", IEEE Trans. Acoust., Speech and Signal Processing, Vol. ASSP-26, No. 4, pp. 284-290, August 1978.

[Cooley1965] J. W. Cooley and J. W. Tukey, "An Algorithm for the Machine Calculation of Complex Fourier Series", Mathematics Computation, Vol. 19, pp. 297–301, April 1965.

[Cui2005] T. Cui and C. Tellambura, "An efficient generalized sphere decoder for rank-deficient MIMO systems", IEEE Communications Letters, Vol. 9, No.5, pp.423-425, May 2005.

[Daher2008] A. Daher, E. H. Baghious and G. Burel, "Fast Algorithm for optimal Design of Block Digital Filters Based on Circulant Matrices ", IEEE Signal Processing Letters, Vol. 15, 2008.

[Daher2009-a] A. Daher, E. H. Baghious, G. Burel and E. Radoi, "Overlap-Save and Overlap-Add Filters : Optimal Design and Comparison", IEEE Transactions on Signal Processing, (Accepted).

[Daher2009-b] A. Daher, E. H. Baghious and G. Burel, "Optimal generalized design of transform-based block digital filters" 2009 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2009), pp.3193-3196, April 19-24, 2009, Taipei, Taiwan.

[Damen2000] M.O. Damen, K. Abed-Meraim, and J.C. Belfiore, "Generalized sphere decoder for asymmetrical space-time communication architecture", IEEE Electronics Letters, Vol. 36, No. 2, pp. 166-167, January 2000.

[Damen2003] M.O. Damen, H. El Gamal, and G. Caire, "On maximum-likelihood detection and the search for the closest lattice point", IEEE Transactions on Information Theory, Vol. 49, No. 10, pp.2389-2402, October 2003.

[Dayal2003] P. Dayal and M. K. Varanasi, "A fast generalized sphere decoder for optimum decoding of under-determined MIMO systems", In Proceedings of 41st Annu. Allerton Conf. Communication, Control, and Computing, pp. 1216-1225, Monti- cello, IL, October 2003.

[Duhamel1982] P. Duhamel andH. Hollmann, "Number Theoretic Transforms with 2 as root of unity", Electronics letters, Vol. 18, No. 22, pp. 978-980, October 1982.

[Fincke1985] U. Fincke and M. Pohst, "Improved methods for calculating vectors of short length in a lattice, including a complexity analysis", Mathematics of Computation, Vol 44, pp. 463-471, April 1985.

[Foschini1996] G. J. Foschini, "Layered space-time architecture for wireless communication in a fading environment when using multi-elements antennas", Bell Labs, Tech. J., Vol. 1, No. 2, pp. 41-59, 1996.

[Frantz2004] G. Frantz and R. Simar, "Comparing Fixed- and Floating-Point DSPs", Texas Instruments, Dallas, TX, USA, 2004. White paper available at *ocus.ti.com/lit/ml/spry061/spry061.pdf.*

[Gold1969] B. Gold and K. L. Jordan, "A direct search procedure for designing finite duration impulse response filters", IEEE Trans. Audio Electroacoust., Vol. AU-17, No. 1, pp. 33-36, March 1969.

[Golub1989] G. H. Golub and C. F. Van Loan, Matrix Computations, Second ed. Baltimore, MD: The John Hopkins Univ. Press, 1989.

[Gray2006] R. M. Gray, Toeplitz and Circulant Matrices: A review. Now Publishers Inc., 2006, ISBN: 1-933019-23-9.

[Grotschel1993] M. Grotschel, L. Lovasz, and A. Schriver, Geometric Algorithms and Combinatorial Optimization. Springer Verlag, 2nd ed. 1993.

[Hassibi2005] B. Hassibi and H. Vikalo, "On the sphere-decoding algorithm I : Expected complexity", IEEE Transactions on Signal Processing, Vol. 53, No. 8, pp. 2806–2818, 2005.

[Helms1971] H. D. Helms, "Digital filters with equiripple or minimax responses", IEEE Trans. Audio Electroacoust.. vol. AU-19. No.1, pp. 87-93, March 1971.

[Julien1991] G. A. Julien, "Number Theoretic Techniques in Digital Signal Processing", Academic Press, Vol. 80, Chap. 2, pp. 69-163, 1991.

[Kaiser1966] J. F. Kaiser and F. F. Kuo, System Analysis by Digital Computer. New York: Wiley. 1966, ch. 7.

[Kaiser1974] J. F. Kaiser, "Nonrecursive digital filter design using the I,-sinh window functions ", in Proc. 1974 IEEE Svmp. Circuits Syst. , pp. 20-23, April 1974.

[Kannan1983] R. Kannan, " Improved algorithms for integer programming and related lattice problems ". In Proceedings of ACM Symposium on Theory of Computing, pp. 19-206, April 1983.

[Kannan1987] R. Kannan, " Minkowski's convex body theorem and integer programming", Mathematics of Operations Research, Vol. 12, No. 3, pp. 415-440, August 1987.

[Kaufman1977] A. Kaufman and A. Henry-Labordere, Integer and Mixed Programming Theory and Applications. New York : Academic, 1977.

[Kellogg1972] W. C. Kellogg, "Time domain design of nonrecursive least mean square digital filters", IEEE Trans. Audio Electroacoust., Vol. AU-20, No. 2, pp. 155-158, June 1972.

[Knuth1969] D. E. Knuth, "The Art of Computer Programming", Vol. 2 : Seminumerical Algorithms, Addison-Wesley, 1969.

[Kodek1980] D.M. Kodek, "Design of Optimal Finite Wordlength FIR Digital Filters Using Integer Programming Techniques", IEEE Trans. Acoust. Speech, Signal Process., Vol. ASSP-28, No. 3, pp. 304-308, June 1980.

[Lagarias1990] J. Lagarias, H. Lenstra, C. Schnorr, "Korkin-Zolotarev bases and successive minima of a lattice and its reciprocal lattice", Combinatorica, Vol. 10, pp. 333-348, 1990.

[Leibowitz1976] L. M. Leibowitz, "A simplified binary arithmetic for the Fermat number transform", IEEE Trans. Acoust., Speech and Signal Process., Vol. ASSP-24, No. 5, pp. 356-359, October 1976.

[Lenstra1982] A. Lenstra, H. Lenstra, and L. Lovasz, " Factoring polynomials with rational coefficients", Mathematische Annalen, Vol. 261, No. 4, pp. 515-534, December 1982.

[Lernon1999] D.Mc Lernon, "One dimensional LPTV structures : derivations, interrelationships and properties" , IEE Proc. Image Signal Process., Vol 146, No 5, pp. 245-252, October 1999.

[Lim1983] Y. C. Lim and S. R. Parker, "Discrete coefficient FIR digital filter design based upon an LMS criteria", IEEE Trans. Circuits Syst., Vol. CAS-30, No. 10, pp. 723-739, October 1983.

[Loeffler1984] C. M. Loeffler and C. S. Burrus "Optimal design of periodically time-varying and multirate digital filters", EEE transactions on acoustics, speech, and signal processing ISSN 0096-3518, Vol. 32, No 5, pp. 991-997, October 1984.

[Luo2003] Z.-Q. Luo, X. Luo, and M. Kisialiou, "An efficient quasi-maximum likelihood decoder for PSK signals", In Proceedings of IEEE International Conference on ASSP, Vol 6, pp. 561-564, April 2003.

[Malvar1992] H. S. Malvar, Signal Processing With Lapped Transforms, Reading, MA: Artech House, 1992.

[McClellan1973] J. H. McClellan and T. Parks, "A unified approach to the design of optimum FIR linear-phase digital filters", IEEE Trans. Circuit Theory, vol. CT-20, pp. 697-701, Nov. 1973.

[McClellan1976] J. H. McClellan, "Hardware Realization of a Fermat Number Transform", IEEE Trans. Acoust., Speech and Signal Process., Vol. ASSP-24, No. 3, pp. 216-225, June 1976.

[Micciancio2001] D. Micciancio, "The hardness of the closest vector problem with preprocessing" IEEE Transactions on Information Theory, Vol. 47, No. 3, pp.1212-1215, Mach 2001.

[Mobasher2005] A. Mobasher, M. Taherzadeh, R. Sotirov, and A. K. Khandani, "An efficient quasi-maximum likelihood decoding for finite constellations", In Proceedings of Information Sciences and Systems Conference, The Johns Hopkins University, March 2005.

[Mobasher2007] A. Mobasher and A. K. Khandani, "Matrix-Lifting Semi-Definite Programming for Decoding in Multiple Antenna Systems", Information Theory, 2007, CWIT '07. 10th Canadian Workshop on, ISBN : 1-4244-0769-9, pp. 136-139,June 2007.

[Nussbaumer1976] H.J. Nussbaumer, "Digital Filtering Using Complex Mersenne Transforms", IBM J. Res. Develop. 20, pp. 498-504, 1976.

[Nussbaumer1977] H.J. Nussbaumer, "Digital Filtering Using Pseudo- Mersenne Transforms", IEEE Trans. Acoust., Speech and Signal Process., Vol. ASSP-25, pp. 79-83, August 1977.

[Nussbaumer1978] H.J. Nussbaumer, "Relative Evaluation of Various Number Theoretic Transforms for filtering Applications", IEEE Trans. Acoust., Speech and Signal Process., Vol. ASSP-26, No. 1, pp. 88-93, August 1978.

[Oppenheim1975] A. V. Oppenheim and R. W. Schafer, Digital Signal Processing, Englewood Cliffs, NJ: Prentice-Hall, Inc., 1975.

[Oppenheim1989] A. V. Oppenheim and R. W. Schafer, Discrete-Time Signal Processing. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[Papoulis1962] A. Papoulis, The Fourier Integral and Its Applications. NewYork: Mc- Graw-Hill, 1962.

[Parks1972] [1] T.W. Parks and J.H. McClellan, "Chebyshev approximation for nonrecursive digital filters with linear phase'', IEEE Trans. Circuit Theory, vol. CT-19, No. 2, pp. 189–194, Mach 1972.

[Parks1987] T. W. Parks and C. S. Burrus, Digital Filter Design, New York: John Wiley and Sons, Inc., June 1987.

[Pohst1981] M. Pohst. On the computation of lattice vectors of minimal length, successive minima and reduced bases with applications. ACM SIGSAM Bulletin, Vol. 15, pp. 37-44, February 1981.

[Rabiner1970] L. R. Rabiner, B. Gold,. and C. A. McGonegal, "An approach to the approximation problem for nonrecursive digital filters", IEEE Trans. Audio Electroacoust., vol. AU-18, No. 2, pp. 83-106, June 1970.

[Rabiner1971] L. R. Rabiner, "Techniques for designing finite-duration impulse response digital filters", IEEE Trans. Commun. Technol., vol. COM-19, No. 2, pp. 188-195, Apr. 1971.

[Rabiner1975] L. R. Rabiner and B. Gold, Theory and Application of Digital Signal Processing, Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1975, ISBN 0139141014.

[Rader1972] C. M. Rzder, "Discrete Convolutions via Mersenne Transrorms", IEEE Transactions on computers, Vol. C-21, No. 12, pp. 1269-1273, December 1972.

[Remez1934] E. Ya. Remez, "Sur la détermination des polynômes d'approximation de degré donnée", Comm. Soc. Math. Kharkov, Vol. 10, pp. 41-63, 1934.

[Rosen1993] K. H. Rosen, "Elementary Number Theory and its Applications", Third Edition, Addison-Wesley, 1993.

[Salkin1975] H. M. Salkin, Integer Programming. Reading, MA : Addison-Wesley, 1975.

[Schnorr1994] C.P. Schnorr and M. Euchner, "Lattice basis reduction : improved practical algorithms and solving subset sum problems" Mathematical Programming, vol. 66, pp. 181-199, 1994.

[Tan2001] P. Tan and L.K. Rasmussen, "The application of semidefinite programming for detection in CDMA", IEEE Journal on Selected Areas in Communications, Vol 19, No. 8, pp. 9, August 2001.

[Teunissen1993] P.J.G. Teunissen, "Least-squares estimation of the integer GPS ambiguities", In Proceedings of Section IV Thery and Methodology, IAG General Meeting, Beijing, China, pp. 561-571, August 1993.

[Teunissen1995-a] P.J.G. Teunissen "The invertible GPS ambiguity transformation", Manuscripta Geodatetica, Vol. 20, No. 6, pp. 489-497, 1995.

[Teunissen1995-b] P.J.G. Teunissen, "The least-squares ambiguity decorrelation adjustment : a method for fast GPS ambiguity estitmation", Journal of Geodesy, Vol. 70, No. 1-2, pp. 65-82, 1995.

[Teunissen1999] P.J.G. Teunissen, "An optimality property of the integer least-squares estimator", Journal of Geodesy, Vol. 73, No. 11, pp. 587-593, 1999.

[Tufts1970] D. W. Tufts and J. T. Francis, "Designing digital lowpass filters: Comparison of some methods and criteeria", IEEE Trans. Audio Electroucoust., vol. AU-18, pp. 487-494, December 1970.

[Yang2005] Z. Yang, C. Liu, and J. He, "A new approach for fast generalized sphere decoding in MIMO systems", IEEE Signal Processing Letters, Vol. 12, No. 1, pp. 41-44, January 2005.

[Winograd1980] S. Winograd, Arithmetic complexity of computation, CBMS-NSF Regional Conference Series in Applied Mathematics, SIAM publications, No. 33, 1980.

[Wong1991] P. W. Wong, "Quantization and roundoff noises in fixed-point FIR digital filters", IEEE Transactions on Signal Processing, Vol. 39, No. 7, pp. 1552-1563, July 1991.

[Wubben2001] D. Wubben, R. Bohnke, J. Rinas, V. Kuhn, and K.D. Kammeyer, "Efficient algorithm for decoding layered space-time codes", IEEE Electronics Letters, Vol 37, No. 22, pp. 1348-1350, October 2001.

Liste des publications

Publications dans des journaux internationaux avec comité de lecture

- A. Daher, E. H. Baghious and G. Burel, "Fast Algorithm for optimal Design of Block Digital Filters Based on Circulant Matrices ", IEEE Signal Processing Letters, Vol. 15, 2008.
- A. Daher, E. H. Baghious, G. Burel and E. Radoi, "Overlap-Save and Overlap-Add Filters : Optimal Design and Comparison", IEEE Transactions on Signal Processing, (Accepté).
- A. Daher, E. H. Baghious, E. Radoi, G. Burel and R. Gautier, "Optimal Design of Fermat Number Transform-Based Block Digital Filters Using a Quadratic Criterion" (En cours de soumission).

Communications internationales avec comité de lecture

- A. Daher, E. H. Baghious and G. Burel, "Optimal Generalized Design of Transform-Based Block Digital Filters", pp.3193-3196, 2009 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2009), April 19-24, 2009, Taipei, Taiwan.

Résumé : L'objectif principal de notre étude est de développer des algorithmes rapides pour une conception optimale et une implantation de très faible complexité des filtres numériques. Le critère d'optimisation choisi est celui de la minimisation de l'erreur quadratique moyenne. Ainsi, nous avons étudié et développé de nouveaux algorithmes de synthèse des filtres à réponse impulsionnelle finie (RIF) associés aux deux techniques de filtrage par blocs, overlap-save (OLS) et overlap-add (OLA). Ces deux techniques de filtrage RIF consistent à traiter le signal par blocs au moyen de la transformée de Fourier rapide (TFR) et permettent ainsi de réduire la complexité arithmétique des calculs de convolution. Les algorithmes que nous avons proposés sont basés sur le développement du modèle matriciel des structures OLS et OLA et sur l'utilisation des propriétés de l'algèbre linéaire, en particulier celles des matrices circulantes. Pour réduire davantage la complexité et la distorsion de filtrage, nous avons approfondi les bases mathématiques de la transformée en nombres de Fermat (FNT : Fermat Number Transform) qui est amenée à trouver des applications de plus en plus diverses en traitement du signal. Cette transformée, définie sur un corps de Galois d'ordre égal à un nombre de Fermat, est un cas particulier des transformées en nombres entiers (NTT : Number Theoretic Transform). Comparé à la TFR, la FNT permet un calcul sans erreur d'arrondi ainsi qu'une large réduction du nombre de multiplications nécessaires à la réalisation du produit de convolution. Pour mettre en évidence cette transformée, nous avons proposé et étudié une nouvelle conception des filtres blocs OLS et OLA mettant en oeuvre la FNT. Nous avons ensuite développé un algorithme de très faible complexité pour la synthèse du filtre optimal en utilisant les propriétés des matrices circulantes que nous avons développées dans le corps de Galois. Les résultats de l'implantation en virgule fixe du filtrage par blocs ont montré que l'utilisation de la FNT à la place de la TFR permettra de réduire la complexité et les erreurs de filtrage ainsi que le coût de synthèse du filtre optimal.

Mots Clés : ***Filtrage numérique, Filtres à Réponse Impulsionnelle Finie (RIF), Overlap-save, Overlap-add, Matrices circulantes, Implantation virgule fixe, Transformée en nombres entiers (NTT), Transformée en nombres de Fermat (FNT).***

Summary : The main objective of our study is to develop fast algorithms for an optimal design and an implementation with low complexity of digital filters. The optimization criterion is the mean squared error at the filter output. Thus, we have studied and developed new algorithms for synthesis of finite impulse response (FIR) filters related to the two techniques of block filtering, overlap-save (OLS) and overlap-add (OLA). These two filtering techniques consist in processing the signal by blocks and use the fast Fourier transform (FFT) to reduce the complexity of the convolution calculation. Our algorithms, based on the matrix model development of the OLA and OLS structures, use the linear algebra properties, especially those of circulant matrices. To further reduce the complexity and the distortion, we have looked further into the mathematical foundations of the Fermat Number Transform (FNT). This transform is a particular case of the Number Theoretic Transforms (NTT) defined in the Galois field. Compared to the FFT, the FNT allows a calculation without rounding error and a large reduction of the number of multiplications necessary to carry out the convolution product. To highlight this transform, we have proposed and studied a new design of OLS and OLA filtering using the FNT. We have developed a low complexity algorithm for the optimal synthesis of filters using the properties of circulant matrices that we have developed in the Galois field. The simulation results of the block filtering with fixed-point implementation have shown that the use of the FNT instead of the FFT reduces the complexity and the filtering errors, as well as the cost of optimal filter synthesis.

Keywords : ***Digital filtering, Finite Impulse Response (FIR) filters, Overlap-save, Overlap-add, Circulant matrices, Fixed Point Implementation, Number Theoretic Trnasforms (NTT), Fermat Number Transform (FNT).***